

Geometric methods on low-rank matrix and tensor manifolds

André Uschmajew, Bart Vandereycken

Angaben zur Veröffentlichung / Publication details:

Uschmajew, André, and Bart Vandereycken. 2020. "Geometric methods on low-rank matrix and tensor manifolds." In Handbook of variational methods for nonlinear geometric data, edited by Philipp Grohs, Martin Holler, and Andreas Weinmann, 261-313. Cham: Springer.
https://doi.org/10.1007/978-3-030-31351-7_9.

Chapter 9

Geometric Methods on Low-Rank Matrix and Tensor Manifolds



André Uschmajew and Bart Vandereycken

Contents

9.1	Introduction	262
9.1.1	Aims and Outline	263
9.2	The Geometry of Low-Rank Matrices	264
9.2.1	Singular Value Decomposition and Low-Rank Approximation	265
9.2.2	Fixed Rank Manifold	267
9.2.3	Tangent Space	268
9.2.4	Retraction	270
9.3	The Geometry of the Low-Rank Tensor Train Decomposition	271
9.3.1	The Tensor Train Decomposition	273
9.3.2	TT-SVD and Quasi Optimal Rank Truncation	276
9.3.3	Manifold Structure	279
9.3.4	Tangent Space and Retraction	281
9.3.5	Elementary Operations and TT Matrix Format	283
9.4	Optimization Problems	286
9.4.1	Riemannian Optimization	286
9.4.2	Linear Systems	289
9.4.3	Computational Cost	290
9.4.4	Difference to Iterative Thresholding Methods	291
9.4.5	Convergence	293
9.4.6	Eigenvalue Problems	294
9.5	Initial Value Problems	295
9.5.1	Dynamical Low-Rank Approximation	296
9.5.2	Approximation Properties	297
9.5.3	Low-Dimensional Evolution Equations	298
9.5.4	Projector-Splitting Integrator	299
9.6	Applications	302
9.6.1	Matrix Equations	303
9.6.2	Schrödinger Equation	304

A. Uschmajew
Max Planck Institute for Mathematics in the Sciences, Leipzig, Germany
e-mail: uschmajew@mis.mpg.de

B. Vandereycken (✉)
Section of Mathematics, University of Geneva, Geneva, Switzerland
e-mail: bart.vandereycken@unige.ch

9.6.3 Matrix and Tensor Completion	306
9.6.4 Stochastic and Parametric Equations	306
9.6.5 Transport Equations	307
9.7 Conclusions	308
References	308

Abstract In this chapter we present numerical methods for low-rank matrix and tensor problems that explicitly make use of the geometry of rank constrained matrix and tensor spaces. We focus on two types of problems: The first are optimization problems, like matrix and tensor completion, solving linear systems and eigenvalue problems. Such problems can be solved by numerical optimization for manifolds, called Riemannian optimization methods. We will explain the basic elements of differential geometry in order to apply such methods efficiently to rank constrained matrix and tensor spaces. The second type of problem is ordinary differential equations, defined on matrix and tensor spaces. We show how their solution can be approximated by the dynamical low-rank principle, and discuss several numerical integrators that rely in an essential way on geometric properties that are characteristic to sets of low rank matrices and tensors.

9.1 Introduction

The following chapter is an outline of Riemannian optimization and integration methods on manifolds of low-rank matrices and tensors. This field is relatively new. While the minimization of functions or the time evolution of dynamical systems under smooth manifold constraints is of course classical, and can be treated in a quite general context, there are specific peculiarities to sets of low-rank matrices and tensors that make Riemannian methods particularly amenable to these sets in actual algorithms. There are at least two main reasons for this.

The first is that manifolds of low-rank matrices or tensors are images of multilinear maps. This does not only have the advantage of having at hand an explicit global parametrization of the manifold itself, but also provides a simple representation of tangent vectors and tangent space projections by the product rule. The second reason is the singular value decomposition (SVD), which for matrices has the remarkable property of providing metric projections onto the non-convex sets of bounded rank matrices. As we will see, for certain low-rank tensor manifolds the SVD can be of a similar use.

A classical and powerful set of algorithms for handling low-rank constraints for matrices or tensors is based on eliminating the constraints by using the aforementioned multilinear parametrizations, and then optimize the block parameters separately, typically in the form of alternating optimization. In contrast, Riemannian methods try to take advantage of the actual geometry of the image, which for instance can overcome problems of ill-conditioning of the typically non-unique multilinear parametrizations. One of the earlier works where the tangent space

geometry of non-symmetric fixed rank matrices was quite explicitly exploited in numerical algorithms is [59]. It introduced the dynamical low-rank approximation method for calculating low-rank approximations when integrating a matrix that satisfies a set of ordinary differential equations (ODEs), as we will explain in Sect. 9.5.1. In the context of finding rank bounded feasible points for linear matrix inequalities, a similar exploitation of the tangent space for fixed rank symmetric definite matrices already appeared in [84]. For optimization problems with rank constraints, several Riemannian optimization methods were first presented in [79, 98, 113] that each use slightly different geometries of the sets fixed rank matrices. However, all of them show in great detail how the geometry can be exploited in the algorithms, and [98, 113] also include Riemannian Hessians to obtain superlinear convergence. These algorithms fit in the general framework of optimization on manifolds, summarized in the monograph [2], which however does not deal with manifolds of fixed rank matrices. An influential earlier work using geometrical tools close to the subject of this chapter is [45] about the best rank approximation problem for matrices.

The geometric viewpoint on low-rank matrices can be carried over to low-rank tensors as well. Here, some of the main ideas emanated from mathematical physics, specifically spin systems and molecular dynamics which involves low-rank representation of high-dimensional functions [69]. The embedded geometry of tensor train and hierarchical Tucker manifolds has then been worked out in [46, 108] with the goal of providing the tool of Riemannian optimization also to problems of scientific computing and optimization with tensors. Some examples and references for successful application of such methods will be presented in some details later.

9.1.1 *Aims and Outline*

Our aim in this chapter is to provide a high-level overview of the main ideas and tools for optimization and time integration on low-rank manifolds. For this we decided to avoid formal definitions, assumptions or arguments that we considered too technical, and tried to develop the concepts in a more descriptive manner. As a result the chapter contains few rigorous theorems, but the provided references should enable the reader to look up most of the technical details. We also stick to a quite concrete ‘matrix language’ as much as possible and avoid abstract tensor product spaces. In this sense, a tensor will be just an array of numbers, and while this is often sufficient when dealing with practical problems, coordinate-free multilinear algebra can of course be essential for understanding the theoretical foundations, but is out of scope here.

There are several topics that will not be touched at all in this chapter. First of all, for tensors we have restricted to manifolds of tensors with fixed tensor train rank, because it can be quite easily presented. The two other tensor formats that allow for geometric methods in a similar spirit are the Tucker format (related to the multilinear rank) and its hierarchical version, the hierarchical Tucker format.

Another important ignored topic is about the choice of the rank. While we present methods for optimization and integration on manifolds of fixed rank matrices and tensors, the choice of the rank is quite problem dependent and needs to balance the reachable model error with the numerical complexity. This is often achieved adaptively. Of course, if a problem at hand does not allow for a ‘low-rank’ solution in the first place, the methods presented in this chapter are of limited use, albeit still mathematically interesting. Finding conditions that ensure low-rank solutions to a class of optimization problems or ODEs can be challenging and several questions in this context are still unanswered, especially for tensors.

Finally, the alternating optimization methods mentioned above, like the alternating least squares or DMRG algorithm, will not be further discussed in this chapter. Compared to Riemannian optimization, these classic approaches to low-rank optimization are much better known and have been used in countless applications. For further reading we would like to refer to the several overview articles taking different perspectives on low-rank optimization, see [6, 15–17, 37, 61, 100], and the monographs [39, 51, 53].

The chapter is structured as follows. In Sect. 9.2 we provide an elementary outline of the geometry of the set of fixed rank matrices as an embedded submanifold with focus on the geometric concepts that are needed in efficient algorithms. In Sect. 9.3 we introduce the tensor train format and show that its geometry shares many similarities to that of the matrix case. The next two Sects. 9.4 and 9.5, are devoted to optimization problems and the integration of ODEs over low-rank matrices and tensor train tensors. In both cases we will show how the geometry that was just derived plays a crucial role. Finally, in Sect. 9.6 we mention typical applications that can be treated well with low-rank tensor techniques and in particular with geometric methods.

9.2 The Geometry of Low-Rank Matrices

As motivated in the introduction, many approximation and identification problems involving low-rank matrices or tensors can be formulated as nonlinear, rank constrained optimization problems. To design and understand efficient geometric methods for their solution, it is therefore necessary to understand the geometry of sets of matrices and tensors of bounded rank. The most basic ingredients for such methods are the representation of tangent vectors, the computation of tangent space projections and the availability of retractions. In this section we present these concepts for the well known case of low-rank matrices in quite some detail as it features all the core ideas on an easily understandable level. We will then in the next section consider manifolds of tensors in low rank tensor train format as an exemplary case for tensors, since it is a tensor decomposition with many parallels to the matrix case.

We restrict the considerations to the linear space $\mathbb{R}^{m \times n}$ of real $m \times n$ matrices, although most of the following theory can be developed for complex matrices too.

The Euclidean structure of this space is given by the *Frobenius inner product* of two matrices,

$$(X, Y)_F = \text{trace}(X^T Y) = \sum_{i_1=1}^m \sum_{i_2=1}^n X(i_1, i_2)Y(i_1, i_2),$$

which induces the *Frobenius norm* $\|X\|_F = (X, X)_F^{1/2}$.

As is well known, the *rank* of a matrix $X \in \mathbb{R}^{m \times n}$ is the smallest number $r = \text{rank}(X)$ such that there exist a decomposition

$$X = GH^T, \quad G \in \mathbb{R}^{m \times r}, \quad H \in \mathbb{R}^{n \times r}. \quad (9.1)$$

Necessarily, it holds $r \leq \min(m, n)$. We call such a rank revealing decomposition of X the (G, H) -format.

Note that the decomposition (9.1) is not unique, since we may replace G with GA and H with HA^{-T} , where A is an invertible $r \times r$ matrix. This ambiguity can be removed by requiring additional constraints. A special case is the rank revealing QR decomposition $X = QR$, where $Q \in \mathbb{R}^{m \times r}$ has pairwise orthonormal columns, and $R \in \mathbb{R}^{r \times n}$ is an upper triangular matrix with positive diagonal entries. Such a decomposition can be computed by the column pivoted QR algorithm; see [35].

When m or n are very large, but r is small, it is obviously beneficial in computations to store the matrix X in the (G, H) -format (9.1): instead of storing mn entries of the full matrix X , we only need to know the $(m+n)r$ entries of the matrices G and H . When $(m+n)r$ is much smaller than mn , we may rightfully say that X is of *low rank*. The key idea of low-rank approximation is that in many applications X may not be of exact low rank, but still can be well approximated by a low-rank matrix.

9.2.1 Singular Value Decomposition and Low-Rank Approximation

The fundamental tool for low-rank approximation is the *singular value decomposition* (SVD). Let $\text{rank}(X) \leq r \leq \min(m, n)$, then the SVD of X is a decomposition

$$X = U \Sigma V^T = \sum_{\ell=1}^r \sigma_\ell u_\ell v_\ell^T, \quad (9.2)$$

where $U = [u_1 \cdots u_r] \in \mathbb{R}^{m \times r}$ and $V = [v_1 \cdots v_r] \in \mathbb{R}^{n \times r}$ have orthonormal columns and $\Sigma \in \mathbb{R}^{r \times r}$ is a diagonal matrix. Its diagonal entries $\sigma_1, \dots, \sigma_r$ are called the *singular values* of X and will always be taken to be nonnegative and

ordered: $\sigma_1 \geq \dots \geq \sigma_r \geq 0$. Note that if $k = \text{rank}(X) < r$, then $\sigma_k > 0$, while $\sigma_{k+1} = \dots = \sigma_r = 0$.

The discovery of the SVD is usually attributed to Beltrami and Jordan around 1873/1874, with important later contributions by Sylvester, Schmidt, and Weyl; see, e.g., [104] for a history. Its existence is not difficult to show when appealing to the spectral theorem for symmetric matrices. It is enough to consider $r = \text{rank}(X)$. The positive semidefinite matrix XX^T then has r positive eigenvalues and admits an eigenvalue decomposition $XX^T = U\Lambda U^T$ with $\Lambda \in \mathbb{R}^{r \times r}$ being a diagonal matrix with a positive diagonal, and $U^T U = I_r$. The matrix UU^T is then the orthogonal projector on the column space of X , and hence $UU^T X = X$. Now setting $\Sigma = \Lambda^{1/2}$ and $V = X^T U \Sigma^{-1}$ we obtain $U\Sigma V^T = UU^T X = X$, that is, an SVD of X . Note that V indeed has orthonormal columns, as $V^T V = \Sigma^{-1} U^T X X^T U \Sigma^{-1} = \Sigma^{-1} \Lambda \Sigma^{-1} = I_r$.

The following theorem is the reason for the importance of the SVD in modern applications involving low rank approximation of matrices and—as we will explain later—of tensors.

Theorem 9.1 *Consider an SVD (9.2) of a matrix X with $\sigma_1 \geq \dots \geq \sigma_r \geq 0$. For any $k < r$, the truncated SVD*

$$X_k = \sum_{\ell=1}^k \sigma_\ell u_\ell v_\ell^T$$

provides a matrix of rank at most k that is closest in Frobenius norm to X . The distance is

$$\|X - X_k\|_F = \min_{\text{rank}(Y) \leq k} \|X - Y\|_F = \left(\sum_{\ell=k+1}^r \sigma_\ell^2 \right)^{1/2}. \tag{9.3}$$

If $\sigma_k > \sigma_{k+1}$, then X_k has rank k and is the unique best approximation of rank at most k .

This famous theorem is due to Schmidt [96] dating 1907 who proved it for compact integral operators. Later in 1936 it was rediscovered by Eckart and Young [25]. In 1937, Mirksy [80] proved a much more general version of this theorem stating that the same truncated SVD provides a best rank- k approximation in any unitarily invariant norm. A norm $\|\cdot\|$ on $\mathbb{R}^{m \times n}$ is called unitarily invariant if $\|X\| = \|QXP\|$ for all orthogonal Q and P . For such a norm it holds that $\|X\| = \|\Sigma\|$, that is, the norm is entirely defined by the vector of singular values.

The SVD of an $m \times n$ matrix can be computed from a symmetric eigenvalue problem or, better, using the Golub–Kahan algorithm [34]. The amount of work

in double precision¹ when $m \geq n$ is $O(14mn^2 + 8n^3)$; see [35, Chapter 8.6]. For a large matrix X , computing the full SVD is prohibitively expensive if one is only interested in its low-rank approximation X_k and if $k \ll \min(m, n)$. To this end, there exist many so-called matrix-free methods based on Krylov subspaces or randomized linear algebra; see, e.g., [43, 67]. In general, these methods are less predictable than the Golub–Kahan algorithm and are not guaranteed to always give (good approximations of) X_k . They can, however, exploit sparsity since they only require matrix vector products with X and X^T .

Observe that the existence of a best approximation of any matrix X by another matrix of rank at most k implies that the set

$$\mathcal{M}_{\leq k} = \{X \in \mathbb{R}^{m \times n} : \text{rank}(X) \leq k\} \quad (9.4)$$

is a closed subset of $\mathbb{R}^{m \times n}$. Therefore any continuous function $f: \mathbb{R}^{m \times n} \rightarrow \mathbb{R}$ with bounded sublevel sets attains a minimum on $\mathcal{M}_{\leq k}$. The formula (9.3) for the distance from $\mathcal{M}_{\leq k}$ implies that a matrix admits a good low-rank approximation in Frobenius norm if its singular values decay sufficiently fast. Consequently, low-rank optimization is suitable for such matrix problems, in which the true solution can be expected to have such a property.

9.2.2 Fixed Rank Manifold

Geometric optimization methods, like the ones we will discuss later, typically operate explicitly on smooth manifolds. The set $\mathcal{M}_{\leq k}$ of matrices of rank at most k is a real algebraic variety, but not smooth in those points X of rank strictly less than k . The good news is that the set $\mathcal{M}_{\leq k-1}$ of these points is of relative Lebesgue measure zero.

The *smooth part* of the variety $\mathcal{M}_{\leq k}$ is the set

$$\mathcal{M}_k = \{X \in \mathbb{R}^{m \times n} : \text{rank}(X) = k\}$$

of matrices of fixed rank k . It is a folklore result in differential geometry (see, e.g., [66, Example 8.14]) that \mathcal{M}_k is a C^∞ smooth embedded submanifold of $\mathbb{R}^{m \times n}$ of dimension

$$\dim(\mathcal{M}_k) = mn - (m - k)(n - k) = (m + n - k)k. \quad (9.5)$$

The easiest way to show this is by explicitly constructing \mathcal{M}_k as the union of level sets of submersions. The idea is as follows.

¹Like for eigenvalues, computing the SVD has to be done iteratively and hence will not terminate in finite time in exact arithmetic for a general matrix.

We partition the matrices in $\mathbb{R}^{m \times n}$ as

$$X = \begin{bmatrix} A & B \\ C & D \end{bmatrix}, \quad A \in \mathbb{R}^{k \times k},$$

and consider the open set \mathcal{U} of all matrices, for which block A is invertible. A matrix X in \mathcal{U} then has rank k if and only if the Schur complement $F(X) = D - CA^{-1}B$ vanishes, that is, $\mathcal{M}_k \cap \mathcal{U} = F^{-1}(0)$. The function F is a submersion from \mathcal{U} to $\mathbb{R}^{(m-k) \times (n-k)}$ because it is surjective (consider $B = C = 0$), and its partial derivative at any point $X \in \mathcal{U}$ with respect to D is the identity, hence the derivative $F'(X)$ at X is surjective. By the submersion theorem, the above preimage $\mathcal{M}_k \cap \mathcal{U}$ is therefore an embedded submanifold of the specified dimension (9.5), and it remains to note that the full set \mathcal{M}_k is the finite union of such manifolds $\mathcal{M}_k \cap \mathcal{U}$ over all possible positions of a $k \times k$ invertible submatrix A .

As an alternative to the above proof, \mathcal{M}_k can also be described as a smooth quotient manifold as in [82]; see also [1] for an overview.

Another important remark concerning optimization is that for $k < \min(m, n)$ both the sets \mathcal{M}_k and $\mathcal{M}_{\leq k}$ are simply connected. This follows from the rank revealing decomposition (9.1) and the connectivity of non-singular k frames in \mathbb{R}^n .

9.2.3 Tangent Space

The explicit knowledge of the tangent spaces and the efficient representation of tangent vectors is crucial for the practical implementation of geometric optimization methods on a manifold. For the fixed rank manifold we have several options for representing tangent vectors.

First of all, it follows from the bilinearity of the map $(G, H) \mapsto GH^T$ that matrices of the form

$$\xi = \dot{G}H^T + G\dot{H}^T, \quad \dot{G} \in \mathbb{R}^{m \times k}, \quad \dot{H} \in \mathbb{R}^{n \times k}, \quad (9.6)$$

are tangent vectors to \mathcal{M}_k at $X = GH^T$. Like the (G, H) -format, this representation of tangent vectors has the disadvantage of not being unique, and it might be sensitive to numerical errors when G or H are ill conditioned.

On the other hand, the representation (9.6) reveals that the tangent vector ξ lies in the sum of two overlapping linear spaces, namely, the subspaces of all matrices whose column (resp. row) space is contained in the column (resp. row) space of X . Based on this observation we can find another representation for ξ . Let $U \in \mathbb{R}^{m \times k}$ and $V \in \mathbb{R}^{n \times k}$ contain orthonormal bases for the column and row space of $X \in \mathcal{M}_k$. Then $X = USV^T$ for some $S \in \mathbb{R}^{k \times k}$ (a possible choice here is the SVD (9.2) of

X , that is, $S = \Sigma$). We choose corresponding orthonormal bases $U_{\perp} \in \mathbb{R}^{m \times (m-k)}$ and $V_{\perp} \in \mathbb{R}^{n \times (n-k)}$ for the orthogonal complements. Then the tangent vector ξ is an element of the linear space

$$T_X \mathcal{M}_k = \left\{ [U \ U_{\perp}] \begin{bmatrix} C_{11} & C_{12}^T \\ C_{21} & 0 \end{bmatrix} [V \ V_{\perp}]^T : \right. \\ \left. C_{11} \in \mathbb{R}^{k \times k}, C_{21} \in \mathbb{R}^{(m-k) \times k}, C_{12} \in \mathbb{R}^{(n-k) \times k} \right\}. \quad (9.7)$$

Vice versa, it is not too difficult to show that every element in $T_X \mathcal{M}_k$ can be written in the form (9.6) and hence is a tangent vector. Since the dimension of $T_X \mathcal{M}_k$ equals that of \mathcal{M}_k , it follows that in fact $T_X \mathcal{M}_k$ is equal to the tangent space to \mathcal{M}_k at X .

In (9.7) we have decomposed the tangent space $T_X \mathcal{M}_k$ into three mutually orthogonal subspaces represented by the three matrices C_{11} , C_{21} and C_{12} . The orthogonal projection of any matrix $Z \in \mathbb{R}^{m \times n}$ onto $T_X \mathcal{M}_k$ is hence obtained by projecting on these three spaces separately. This gives

$$\mathcal{P}_X(Z) = P_U Z P_V + (I - P_U) Z P_V + P_U Z (I - P_V), \quad (9.8)$$

where $P_U = U U^T$ and $P_V = V V^T$ are the orthogonal projections onto the column and row space of X , respectively. Expanding this expression, gives the alternative formula

$$\mathcal{P}_X(Z) = P_U Z + Z P_V - P_U Z P_V. \quad (9.9)$$

While the characterization (9.7) of $T_X \mathcal{M}_k$ is very convenient for theoretical purposes, it is less suitable in calculations when k is small but m or n are very large, since then also one of the matrices U_{\perp} or V_{\perp} will be very large. In that situation, the factored representation proposed in [98, 113] is preferable:

$$\xi = U \dot{S} V^T + \dot{U} V^T + U \dot{V}^T, \quad (9.10)$$

where

$$\dot{S} = C_{11} \in \mathbb{R}^{k \times k}, \quad \dot{U} = U_{\perp} C_{21} \in \mathbb{R}^{m \times k}, \quad \dot{V} = V_{\perp} C_{12} \in \mathbb{R}^{n \times k}. \quad (9.11)$$

This only requires storing the smaller matrices \dot{S} , \dot{U} , and \dot{V} . Observe that the columns of \dot{U} and \dot{V} are orthogonal to the columns of U and V , respectively, which is also called a *gauging condition*.

To conclude, once U , S and V are chosen to represent $X = USV^T$, all the factored parametrizations of tangent vectors at X belong to the linear subspace²

$$H_{(U,S,V)} = \{(\dot{U}, \dot{S}, \dot{V}) : \dot{S} \in \mathbb{R}^{k \times k}, \dot{U} \in \mathbb{R}^{n \times k}, U^T \dot{U} = 0, \dot{V} \in \mathbb{R}^{m \times k}, V^T \dot{V} = 0\}.$$

The representation of $T_X \mathcal{M}_k$ by $H_{(U,S,V)}$ is bijective. One can therefore directly compute the result of the projection $\mathcal{P}_X(Z)$ as a factored parametrization:

$$\dot{S} = U^T ZV, \quad \dot{U} = (I - P_U)ZV, \quad \dot{V} = (I - P_V)Z^T V. \quad (9.12)$$

Observe that this requires k matrix vector products with Z and Z^T , hence sparsity or a low rank of Z can be exploited nicely.

9.2.4 Retraction

The other main ingredient for efficient geometric methods are retractions. A retraction for a manifold \mathcal{M} is a smooth map R on the tangent bundle $T\mathcal{M}$, and maps at every X the tangent space $T_X \mathcal{M}$ to \mathcal{M} . The decisive property of a retraction is that this mapping is exact to first order, that is,

$$R_X(\xi) = X + \xi + o(\|\xi\|). \quad (9.13)$$

Obviously, such a map will be useful in optimization methods for turning an increment $X + \xi$ on the affine tangent plane to a new point $R_X(\xi)$ on the manifold. For Riemannian manifolds it can be shown that retractions always exist. A very natural way from a differential geometry viewpoint is the so called *exponential map*, which maps along geodesics in direction of the tangent vector. In practice, the exponential map may be very complicated to compute. There are, however, alternative choices. Retractions in our current context³ seem to be first introduced in [99]; see also [2] for more details.

For the embedded submanifold \mathcal{M}_k (more precisely, for $\mathcal{M}_{\leq k}$) we are in the fortunate situation that, by Theorem 9.1, we can compute the metric projection (best approximation) in the ambient space equipped with the Frobenius norm as metric through the truncated SVD. It hence provides an easy-to-use retraction with respect to this metric. Note that in general for a C^m smooth embedded submanifold \mathcal{M} of an Euclidean space with $m \geq 2$ and a point $X \in \mathcal{M}$, there exists an open neighborhood of $0 \in T_X \mathcal{M}$ on which a metric projection $\xi \mapsto \mathcal{P}_{\mathcal{M}}(X + \xi)$ is uniquely defined and satisfies the retraction property

²This subspace is a horizontal distribution for the smooth quotient manifold that factors out the freedom in the parametrization $X = USV^T = (UA)(A^{-1}SB^{-T})(VB)^T$; see [1].

³Not to be confused with a (deformation) retract from topology.

$$\|X + \xi - \mathcal{P}_{\mathcal{M}}(X + \xi)\| = o(\|\xi\|).$$

In addition, $\mathcal{P}_{\mathcal{M}}$ is C^{m-1} smooth on that neighborhood; see, e.g., [68, Lemma 2.1].

When using truncated SVD as a retraction for \mathcal{M}_k , the crucial question arises whether it can be computed efficiently. This indeed is the case. If $X = USV^T$ and $\xi \in T_X \mathcal{M}_k$ are represented in the factored form (9.10), we first compute QR decompositions of \dot{U} and \dot{V} ,

$$\dot{U} = Q_1 R_1, \quad \dot{V} = Q_2 R_2.$$

It then holds

$$X + \xi = \begin{bmatrix} U & \dot{U}_1 \end{bmatrix} \begin{bmatrix} SV^T + \dot{S}V^T + \dot{V}^T \\ V^T \end{bmatrix} = \begin{bmatrix} U & Q_1 \end{bmatrix} K \begin{bmatrix} V & Q_2 \end{bmatrix}^T \tag{9.14}$$

with the $2k \times 2k$ block matrix

$$K = \begin{bmatrix} S + \dot{S} & R_2^T \\ R_1 & 0 \end{bmatrix}.$$

Since the matrices $\begin{bmatrix} U & Q_1 \end{bmatrix}$ and $\begin{bmatrix} V & Q_2 \end{bmatrix}$ each have orthonormal columns (as before we assume that both U and V have orthonormal columns), we can obtain an SVD of the ‘big’ matrix $X + \xi$ from an SVD of the small matrix K , which can be done in $O(k^3)$ time.

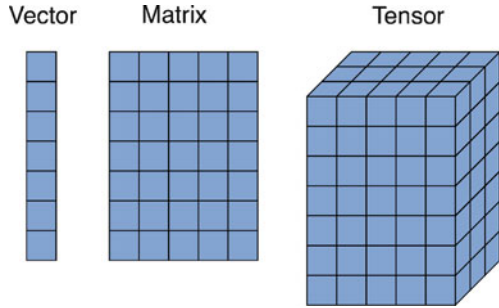
9.3 The Geometry of the Low-Rank Tensor Train Decomposition

In this section we present the tensor train decomposition as a possible generalization of low-rank matrix decomposition to tensors. By tensors we simply mean higher-order analogs of matrices: an $n_1 \times \dots \times n_d$ tensor X is an array of this size containing real valued entries $X(i_1, \dots, i_d)$; see Fig. 9.1. Such data structures appear in many applications. Another way to see them is as multivariate functions depending on discrete variables/indices. The tensors of given size form a linear space denoted as $\mathbb{R}^{n_1 \times \dots \times n_d}$. The number d of directions is called the *order* of the tensor. Matrices are hence tensors of order $d = 2$. As for matrices, it is common to also call the natural Euclidean inner product for tensors,

$$\langle X, Y \rangle_F = \sum_{i_1=1}^{n_1} \dots \sum_{i_d=1}^{n_d} X(i_1, \dots, i_d) Y(i_1, \dots, i_d), \tag{9.15}$$

the Frobenius inner product, and it induces the Frobenius norm.

Fig. 9.1 Tensors of order one (vectors), two (matrices), and three



An $n \times \dots \times n$ tensor has n^d entries, which can quickly become unmanageable in practice when d is large. This is sometimes called a *curse of dimensionality*. Besides other important reasons, the use of low-rank tensor formats provides a tool to circumvent this problem and deal with high dimensional data structures in practice. From a geometric viewpoint a low-rank tensor format defines a nonlinear subset in the space $\mathbb{R}^{n_1 \times \dots \times n_d}$, like the sets $\mathcal{M}_{\leq k}$ from (9.4) in the space of matrices, which can be conveniently represented as the image of a multilinear map. Several choices are possible here.

Let us recall the (G, H) -format (9.1) for a matrix. One way to look at it is as a separation of the variables/indices:

$$X(i_1, i_2) = \sum_{\ell=1}^r G(i_1, \ell)H(i_2, \ell). \tag{9.16}$$

The rank is the minimal number r needed for such a separation. A straightforward analog for tensors would be a decomposition

$$X(i_1, \dots, i_d) = \sum_{\ell=1}^r C_1(i_1, \ell) \cdots C_d(i_d, \ell)$$

with *factor matrices* $C_\mu \in \mathbb{R}^{n_\mu \times r}$, $\mu = 1, \dots, d$. This tensor format is called the *canonical polyadic (CP) format*. The minimal r required for such a decomposition is called the (canonical) *tensor rank* of X . As for matrices, if r is small then storing a tensor in the CP format is beneficial compared to storing all $n_1 \cdots n_d$ entries since one only needs to know the d factor matrices C_1, \dots, C_d .

The CP format has numerous useful applications in data science and scientific computing; see [61] for an overview. One major difference to the matrix case, however, is that the set of all tensors with canonical rank bounded by k is typically not closed. Moreover, while the closure of this set is an algebraic variety, its smooth part is in general not equal to the set of tensors of fixed rank k and does not admit an easy explicit description. An exception is the case of rank-one tensors ($k = 1$): the set of all *outer products* $X = c_1 \circ \dots \circ c_d$, defined by

$X(i_1, \dots, i_d) = c_1(i_1) \cdots c_d(i_d)$, of nonzero vectors $c_\mu \in \mathbb{R}^{n_\mu}$, $\mu = 1, \dots, d$, is an embedded submanifold of dimension $(n_1 + \dots + n_d) - (d - 1)$. (It is indeed a special case of manifolds of fixed tensor train rank to be introduced below.) Riemannian optimization in the CP format is hence possible by considering the d -fold sum of rank-one tensors as a manifold, as proposed in [13]. We will, however, not consider this format further in this chapter. Instead, we will present another way to separate the indices of a tensor, which leads to the tensor train format and yields smooth manifolds more similar to the matrix case.

9.3.1 The Tensor Train Decomposition

The *tensor train* (TT) format of a tensor $X \in \mathbb{R}^{n_1 \times \dots \times n_d}$ can be derived recursively. First, index i_1 is separated from the others, that is,

$$X(i_1, i_2, \dots, i_d) = \sum_{\ell_1=1}^{r_1} G_1(i_1, \ell_1) H_1(\ell_1, i_2, \dots, i_d). \tag{9.17}$$

Note that this is a usual matrix decomposition of the form (9.16) when treating the multi-index (i_2, \dots, i_d) as a single index. Next, in the tensor H_1 the indices (ℓ_1, i_2) are separated from the rest, again by a matrix decomposition,

$$H_1(\ell_1, i_2, \dots, i_d) = \sum_{\ell_2=1}^{r_2} G_2(\ell_1, i_2, \ell_2) H_2(\ell_2, i_3, \dots, i_d), \tag{9.18}$$

yielding

$$X(i_1, i_2, \dots, i_d) = \sum_{\ell_1=1}^{r_1} \sum_{\ell_2=1}^{r_2} G_1(i_1, \ell_1) G_2(\ell_1, i_2, \ell_2) H_2(\ell_2, i_3, \dots, i_d). \tag{9.19}$$

Proceeding in this way, one arrives after d steps at a decomposition of the form

$$X(i_1, \dots, i_d) = \sum_{\ell_1=1}^{r_1} \cdots \sum_{\ell_{d-1}=1}^{r_{d-1}} G_1(i_1, \ell_1) G_2(\ell_1, i_2, \ell_2) \cdots G_{d-1}(\ell_{d-2}, i_{d-1}, \ell_{d-1}) G_d(\ell_{d-1}, i_d), \tag{9.20}$$

with *core tensors* $G_\mu \in \mathbb{R}^{r_{\mu-1} \times n_\mu \times r_\mu}$, $\mu = 1, \dots, d$, and $r_0 = r_d = 1$. (The third dummy mode was added to G_1 and G_d to unify notation.) The core tensors G_1 and G_d are hence just matrices, while G_2, \dots, G_{d-1} are tensors of order three. A decomposition (9.20) is called a *tensor train* or *TT decomposition* of X .

The nested summation in formula (9.20) is in fact a long matrix product. If we denote by $G_\mu(i_\mu)$ the $r_{\mu-1} \times r_\mu$ matrix slices of G_μ , one gets the compact representation

$$X(i_1, \dots, i_d) = G_1(i_1)G_2(i_2) \cdots G_{d-1}(i_{d-1})G_d(i_d) \quad (9.21)$$

of the TT format, which explains the alternative name *matrix product state* (MPS) of this tensor decomposition common in physics. This formula clearly shows the multilinearity of the TT decomposition with respect to the core tensors. Also it is easy to see from (9.21) that a TT decomposition is never unique: we can insert the identity $A_\mu A_\mu^{-1}$ between any two matrix factors to obtain another decomposition. It will turn out below that this group action is essentially the only ambiguity.

In the numerical analysis community, the TT format was developed by Oseledets and Tyrtshnikov in [86, 87] with related formats proposed in [36, 40]. In earlier work, it appeared in theoretical physics under a variety of different forms and names, but is now accepted as MPS; see [97] for an overview.

The number of parameters in the TT decomposition (9.20) is bounded by dnr^2 where $n = \max n_\mu$ and $r = \max r_\mu$. When $r \ll n^{d-2}$, this constitutes a great reduction compared to storing the $n_1 \cdots n_d$ entries in X explicitly. Hence the minimal possible choices for the ‘ranks’ r_μ appearing in the above construction are of interest. The crucial concept in this context is unfoldings of a tensor into matrices.

We define the μ th *unfolding* of a tensor X as the matrix $X^{(\mu)}$ of size $(n_1 \cdots n_\mu) \times (n_{\mu+1} \cdots n_d)$ obtained by taking the partial multi-indices (i_1, \dots, i_μ) as row indices, and $(i_{\mu+1}, \dots, i_d)$ as column indices.⁴ In other words,

$$X^{(\mu)}(i_1, \dots, i_\mu; i_{\mu+1}, \dots, i_d) = X(i_1, \dots, i_d)$$

where the semicolon indicates the separation between the row- and column indices. One can then show the following theorem.

Theorem 9.2 *In a TT decomposition (9.20) it necessarily holds that*

$$r_\mu \geq \text{rank}(X^{(\mu)}), \quad \mu = 1, \dots, d-1. \quad (9.22)$$

It is furthermore possible to obtain a decomposition such that equality holds.

To get an insight into why the above statement is true, first observe that, by isolating the summation over the index j_μ , the TT decomposition (9.20) is in fact equivalent to the simultaneous matrix decompositions

$$X^{(\mu)} = G_{\leq \mu} G_{\geq \mu+1}^T, \quad \mu = 1, \dots, d-1, \quad (9.23)$$

⁴In the following, we silently assume that a consistent ordering of multi-indices is used.

with ‘partial’ TT unfoldings

$$[G_{\leq \mu}(i_1, \dots, i_\mu; \ell_\mu)] = [G_1(i_1) \cdots G_\mu(i_\mu)] \in \mathbb{R}^{n_1 \cdots n_\mu \times r_\mu}$$

and

$$[G_{\geq \mu+1}(i_{\mu+1}, \dots, i_d; \ell_\mu)] = [G_{\mu+1}(i_{\mu+1}) \cdots G_d(i_d)]^T \in \mathbb{R}^{n_{\mu+1} \cdots n_d \times r_\mu}.$$

From (9.23) it follows immediately that the rank condition (9.22) is necessary. Equality can be achieved using the constructive procedure leading to (9.20) with minimal matrix ranks in every step. Let us explain this for the first two steps. Clearly, the first step (9.17) is a rank revealing decomposition of $X^{(1)}$, so the rank of that matrix can be used as r_1 . The minimal admissible r_2 in the second step (9.19) is the rank of the second unfolding $H_1^{(2)}$ of tensor H_1 . Let us show that this rank is not larger than the rank of $X^{(1)}$, and hence both are equal by (9.22). Indeed, if $z = [z(i_3, \dots, i_d)]$ is a vector of length $n_3 \cdots n_d$ such that $X^{(2)}z = 0$ and $y = H^{(2)}z$, then (9.17) yields $0 = \sum_{\ell_1=1}^{r_1} G_1(i_1, \ell_1)y(\ell_1, i_2)$, which implies $y = 0$, since G_1 has rank r_1 . This implies $\text{rank}(H^{(2)}) \leq \text{rank}(X^{(2)})$. One can proceed with a similar argument for the subsequent ranks r_3, \dots, r_d .

Theorem 9.2 justifies the following definition.

Definition 9.3 The vector $\mathbf{r} = (r_1, \dots, r_{d-1})$ with $r_\mu = \text{rank}(X^{(\mu)})$, $\mu = 1, \dots, d - 1$ is called the *TT rank* of a tensor $X \in \mathbb{R}^{n_1 \times \cdots \times n_d}$.

For matrices, the SVD-like decompositions $X = USV^T$ with U and V having orthonormal columns are often particularly useful in algorithms since they provide orthonormal bases for the row and column space. This was for instance important for the projection onto the tangent space $T_X \mathcal{M}_k$ at X , see (9.8) and (9.9). It is possible to impose similar orthogonality conditions in the TT decomposition. Recall, that the TT decomposition of a tensor X is obtained by subsequent rank-revealing matrix decompositions for separating the indices i_1, \dots, i_d one from another. This can actually be done from left-to-right, from right-to-left, or from both directions simultaneously and stopping at some middle index i_μ . By employing QR (resp. LQ) matrix decompositions in every splitting step, it is not so difficult to show that one can find core tensors U_1, \dots, U_{d-1} , as well as V_2, \dots, V_d such that for every μ between 1 and $d - 1$ it holds

$$X^{(\mu)} = U_{\leq \mu} S_\mu V_{\geq \mu+1}^T, \tag{9.24}$$

for some $S_\mu \in \mathbb{R}^{r_\mu \times r_\mu}$, and

$$U_{\leq \mu}^T U_{\leq \mu} = V_{\geq \mu+1}^T V_{\geq \mu+1} = I_{r_\mu}. \tag{9.25}$$

Note that these orthogonality conditions inductively imply that the unfoldings $U_\mu^{(3)}$ as well as $V_\mu^{(1)}$ of core tensors itself have orthonormal columns. In general, for a given μ , we call a TT decomposition with cores $G_v = U_v$ for $v < \mu$, $G_\mu(i_\mu) = U_\mu(i_\mu)S_\mu$ and $G_v = V_v$ for $v \geq \mu + 1$, and satisfying (9.25) a μ -orthogonal TT decomposition of X . It implies (9.24).

One advantage of such a μ -orthogonal TT decomposition is that it provides the orthogonal projections $U_{\leq \mu} U_{\leq \mu}^T$ and $V_{\geq \mu+1} V_{\geq \mu+1}^T$ for the column and row space of $X^{(\mu)}$ in the form of partial TT unfoldings that are hence easily applicable to tensors in TT decomposition. From these projections it will be possible to construct the tangent space projectors to TT manifolds in Sect. 9.3.4.

Note that if a TT decomposition with *some* cores G_1, \dots, G_d is already given, a μ -orthogonal decomposition can be obtained efficiently by manipulating cores in a left-to-right, respectively, right-to-left sweep, where each step consists of elementary matrix operations and QR decompositions and costs $O(dnr^4)$ operations in total. In particular, switching from a μ -orthogonal to a $(\mu + 1)$ - or $(\mu - 1)$ -orthogonal decomposition, only one such step is necessary costing $O(nr^4)$. Observe that the costs are linear in the order d and mode sizes n_μ but fourth-order in the ranks r_μ . In practice, this means the limit for r_μ is about 10^2 to 10^3 , depending on the computing power. We refer to [46, 72, 85] for more details on the implementation and properties of the orthogonalization of TT decompositions.

We conclude with the general remark that algorithmically the TT tensor decomposition is characterized by the concept of *sweeping*, which means that most operations are performed recursively from left-to-right, then right-to-left, and so on. Furthermore, the manipulations on the cores of a TT are based on basic linear algebra. We have already seen that building the decomposition by itself or orthogonalizing a given decomposition can be achieved by a left-to-right sweep involving matrix decompositions only. Next we discuss the important operation of rank truncation that is also achieved in this recursive way.

9.3.2 TT-SVD and Quasi Optimal Rank Truncation

Instead of QR decompositions, one can also use singular value decompositions for constructing a μ -orthogonal TT representation (9.24). One then obtains

$$X^{(\mu)} = U_{\leq \mu} \Sigma_\mu V_{\geq \mu+1}^T \quad (9.26)$$

with $\Sigma_\mu \in \mathbb{R}^{r_\mu \times r_\mu}$ being diagonal. In other words, (9.26) is an SVD of $X^{(\mu)}$.

The advantage of using SVDs for constructing the TT decomposition is that they can be truncated ‘on the fly’, that is, the index splitting decompositions like (9.17) and (9.19) are replaced by truncated SVDs to enforce a certain rank. Specifically, in a left-to-right sweep, at the μ th step, let us assume a partial decomposition

$$\tilde{X}^{(\mu-1)} = U_{\leq \mu-1} H_{\mu-1}$$

with $U_{\leq \mu-1}$ having orthonormal columns has been constructed.⁵ Here we write \tilde{X} , since the tensor may not equal X anymore due to previous rank truncations. The next core U_μ is then obtained from the left singular vectors of a truncated SVD of $H_{\mu-1}^{(2)}$. This procedure is called the *TT-SVD algorithm* [86, 88]. Note that since $U_{\leq \mu-1}$ is orthogonal, the truncated SVD of $H_{\mu-1}^{(2)}$ is implicitly also a truncated SVD of $\tilde{X}^{(\mu)}$.

So if at every step of the TT-SVD algorithm instead of the exact rank r_μ a smaller rank k_μ is used, the result will be a tensor $X_{\mathbf{k}}$ of TT rank (at most) $\mathbf{k} = (k_1, \dots, k_{d-1})$ in d -orthogonal TT format. It now turns out that this result provides a *quasi-optimal* approximation of TT rank at most \mathbf{k} to the initial tensor X . Thus the TT-SVD algorithm plays a similar role for TT tensors as the SVD truncation for matrices.

To state this result, let us define the sets

$$\mathcal{M}_{\leq \mathbf{k}} = \{X \in \mathbb{R}^{n_1 \times \dots \times n_d} : \text{TT-rank}(X) \leq \mathbf{k}\}$$

of tensors of TT rank at most $\mathbf{k} = (k_1, \dots, k_{d-1})$, where the inequality for the rank vector is understood pointwise. By Theorem 9.2, this set is an intersection of low-rank matrix varieties:

$$\mathcal{M}_{\leq \mathbf{k}} = \bigcap_{\mu=1}^{d-1} \{X \in \mathbb{R}^{n_1 \times \dots \times n_d} : \text{rank}(X^{(\mu)}) \leq k_\mu\}. \tag{9.27}$$

Since each of the sets in this intersection is closed, the set $\mathcal{M}_{\leq \mathbf{k}}$ is also closed in $\mathbb{R}^{n_1 \times \dots \times n_d}$. As a result, every tensor X admits a best approximation by a tensor in the set $\mathcal{M}_{\leq \mathbf{k}}$, which we denote by $X_{\mathbf{k}}^{\text{best}}$, that is,

$$\|X - X_{\mathbf{k}}^{\text{best}}\|_F = \min_{\text{TT-rank}(Y) \leq \mathbf{k}} \|X - Y\|_F.$$

The TT-SVD algorithm, on the other hand, can be seen as an alternating projection method for computing an approximation to X in the intersection (9.27).

The following theorem has been obtained in [88].

Theorem 9.4 *Let $X \in \mathbb{R}^{n_1 \times \dots \times n_d}$ have TT rank \mathbf{r} and $\mathbf{k} \leq \mathbf{r}$. Denote by $X_{\mathbf{k}}$ the result of the TT-SVD algorithm applied to X with target rank \mathbf{k} . Let ε_μ be the error in Frobenius norm committed in the μ th truncation step. Then the following estimates hold:*

⁵For consistency we set $U_{\leq 0} = 1$ and $X^{(0)} = H_0 = X$.

$$\|X - X_{\mathbf{k}}\|_F^2 \leq \sum_{\mu=1}^{d-1} \varepsilon_{\mu}^2, \quad (9.28)$$

and

$$\varepsilon_{\mu}^2 \leq \sum_{\ell > k_{\mu}} (\sigma_{\ell}^{\mu})^2 \leq \|X - X_{\mathbf{k}}^{\text{best}}\|_F^2, \quad (9.29)$$

where σ_{ℓ}^{μ} are the singular values of the μ th unfolding $X^{(\mu)}$.

The theorem has two immediate and equally important corollaries. The first of them is that the sequential rank truncation performed by the TT-SVD is, as announced above, a quasi-optimal projection:

$$\|X - X_{\mathbf{k}}\|_F \leq \sqrt{d-1} \|X - X_{\mathbf{k}}^{\text{best}}\|_F. \quad (9.30)$$

The second corollary is a complete characterization of low-rank approximability in the TT format. Since $\|X - X_{\mathbf{k}}^{\text{best}}\|_F \leq \|X - X_{\mathbf{k}}\|_F$, the above inequalities imply

$$\|X - X_{\mathbf{k}}^{\text{best}}\|_F^2 \leq \sum_{\mu=1}^{d-1} \sum_{\ell_{\mu} > k_{\mu}} (\sigma_{\ell_{\mu}}^{\mu})^2.$$

A tensor X will therefore admit good approximation by TT tensors of small rank if the singular values of all the unfoldings $X^{(1)}, \dots, X^{(d-1)}$ decay sufficiently fast to zero. By (9.29) such a decay is also a necessary condition. Similar to the comment on matrix problems, the low-rank TT format is hence suitable in practice for tensor problems where the solution has such a property. Justifying this a-priori can be, however, a difficult task, especially for very large problems, and will not be discussed.

We now sketch a proof of Theorem 9.4. The main argument is the observation that while the best rank- k truncation of a matrix is a nonlinear operation, it is for every input indeed performing a *linear* orthogonal projection that can be realized by multiplying from the left an orthogonal projector onto the subspace spanned by the dominant k left singular vectors of the input. Therefore, before the μ th truncation step, the current μ th unfolding is the result of *some* $\mu - 1$ previous orthogonal projections

$$\tilde{X}^{(\mu)} = \tilde{P}_{\mu-1} \dots \tilde{P}_1 X^{(\mu)}, \quad (9.31)$$

which, however, have all been achieved by a matrix multiplication from the left (since only indices $i_1, \dots, i_{\mu-1}$ have been separated at this point). By comparing to the projected best rank- k_{μ} approximation of $X^{(\mu)}$, it is then easy to prove that $\tilde{X}^{(\mu)}$

has no larger distance (in Frobenius norm) to the set of rank- k_μ matrices than $X^{(k)}$ itself. Hence

$$\varepsilon_\mu \leq \min_{\text{rank}(Y^{(\mu)}) \leq k_\mu} \|X^{(\mu)} - Y^{(\mu)}\|_F \leq \min_{\text{TT-rank}(Y) \leq \mathbf{k}} \|X - Y\|_F = \|X - X_{\mathbf{k}}^{\text{best}}\|_F,$$

where the second inequality is due to (9.27). Since the squared Frobenius distance of $X^{(\mu)}$ to $\mathcal{M}_{\leq k_\mu}$ equals $\sum_{\ell > k_\mu} (\sigma_\ell^\mu)^2$, this proves the second statement (9.29) of the theorem.

Showing the first statement (9.28) is more subtle. One writes $X_{\mathbf{k}}$ as the result of corresponding $d - 1$ orthogonal projections in tensor space:

$$X_{\mathbf{k}} = \mathcal{P}_{d-1} \cdots \mathcal{P}_1 X.$$

The error can then be decomposed into

$$X - X_{\mathbf{k}} = (\mathcal{P}_{d-2} \cdots \mathcal{P}_1 X - \mathcal{P}_{d-1} \cdots \mathcal{P}_1 X) + (X - \mathcal{P}_{d-2} \cdots \mathcal{P}_1 X).$$

The Frobenius norm of the first term is precisely ε_{d-1} . One now has to show that both terms are orthogonal to proceed by induction. Indeed, an easy way to see that for every $\mu = 1, \dots, d - 1$ the result $\mathcal{P}_\mu \cdots \mathcal{P}_1 X$ after the μ th truncation is still in the range of the operator $\mathcal{P}_{\mu-1} \cdots \mathcal{P}_1$ is that the rank truncation of $\tilde{X}^{(\mu)}$ as given by (9.31) may equally be achieved by multiplying from the *right* an orthogonal projector on the dominant k_μ right singular values. Then it is clear that multiplying $\tilde{P}_{\mu-1} \cdots \tilde{P}_1$ from the left again will have no effect.

We conclude with two remarks. The first is that the TT-SVD algorithm can be implemented very efficiently if X is already given in a μ -orthogonal TT decomposition as in (9.24), say, with $\mu = 1$, with moderate TT rank. Then in a left-to-right sweep it is sufficient to compute SVDs of single cores, which is computationally feasible if ranks are not too large. This is important in practice when using the TT-SVD algorithm as a retraction as explained below.

The second remark is that the target ranks in the TT-SVD procedure can be chosen adaptively depending on the desired accuracies ε_μ . Thanks to Theorem 9.4 this gives full control of the final error. In this scenario the algorithm is sometimes called *TT-rounding* [86].

9.3.3 Manifold Structure

It may appear at this point that it is difficult to deal with the TT tensor format (and thus with its geometry) computationally, but this is not the case. Tensors of low TT rank can be handled very well by geometric methods in a remarkably analogous way as to low-rank matrices. To do so, one first needs to reveal the geometric structure.

Similar to matrices, the set $\mathcal{M}_{\leq \mathbf{k}}$ of tensors of TT rank bounded by $\mathbf{k} = (k_1, \dots, k_{d-1})$ is a closed algebraic variety but not a smooth manifold. Let us assume that the set of tensors of *fixed* TT rank \mathbf{k} , that is, the set

$$\mathcal{M}_{\mathbf{k}} = \{X \in \mathbb{R}^{n_1 \times \dots \times n_d} : \text{TT-rank}(X) = \mathbf{k}\},$$

is not empty (the conditions for this are given in (9.32) below). Based on Theorem 9.2 it is then easy to show that $\mathcal{M}_{\mathbf{k}}$ is relatively open and dense in $\mathcal{M}_{\leq \mathbf{k}}$. One may rightfully conjecture that $\mathcal{M}_{\mathbf{k}}$ is a smooth embedded manifold in $\mathbb{R}^{n_1 \times \dots \times n_d}$. Note that while $\mathcal{M}_{\mathbf{k}}$ is the intersection of smooth manifolds (arising from taking the conditions $\text{rank}(X^{(\mu)}) = k_{\mu}$ in (9.27)), this by itself does not prove that $\mathcal{M}_{\mathbf{k}}$ is a smooth manifold.

Instead, one can look again at the global parametrization $(G_1, \dots, G_d) \mapsto X$ of TT tensors given in (9.20) but with ranks k_{μ} . This is a multilinear map τ from the linear parameter space $\mathcal{W}_{\mathbf{k}} = \mathbb{R}^{k_0 \times n_1 \times k_1} \times \dots \times \mathbb{R}^{k_{d-1} \times n_d \times k_d}$ (with $k_0 = k_d = 1$) to $\mathbb{R}^{n_1 \times \dots \times n_d}$ and its image is $\mathcal{M}_{\leq \mathbf{k}}$. One can now show that the condition $\text{TT-rank}(X) = \mathbf{k}$ is equivalent to the conditions $\text{rank}(G_{\mu}^{(1)}) = k_{\mu-1}$ and $\text{rank}(G_{\mu}^{(2)}) = k_{\mu}$ on the unfoldings of core tensors, which defines a subset $\mathcal{W}_{\mathbf{k}}^*$ of parameters. The conditions

$$k_{\mu-1} \leq n_{\mu} k_{\mu}, \quad k_{\mu} \leq n_{\mu} k_{\mu-1}, \quad \mu = 1, \dots, d, \tag{9.32}$$

are necessary and sufficient for the existence of such cores, and hence for $\mathcal{M}_{\mathbf{k}}$ being non-empty. Given these conditions the set $\mathcal{W}_{\mathbf{k}}^*$ is open and dense in $\mathcal{W}_{\mathbf{k}}$ and its image under τ is $\mathcal{M}_{\mathbf{k}}$. Yet this parametrization is not injective. From the compact matrix product formula (9.21), we have already observed that the substitution

$$G_{\mu}(i_{\mu}) \rightarrow A_{\mu-1}^{-1} G_{\mu}(i_1) A_{\mu}, \tag{9.33}$$

where A_{μ} are invertible $r_{\mu} \times r_{\mu}$ matrices, does not change the resulting tensor X . One can show that this is the only non-uniqueness in case that X has exact TT rank \mathbf{k} , basically by referring to the equivalence with the simultaneous matrix decompositions (9.23). After removing this ambiguity by suitable gauging conditions, one obtains a locally unique parametrization of $\mathcal{M}_{\mathbf{k}}$ and a local manifold structure [46].

An alternative approach, that provides a global embedding of $\mathcal{M}_{\mathbf{k}}$, is to define an equivalence relation of equivalent TT decompositions of a tensor $X \in \mathcal{M}_{\mathbf{k}}$. The equivalence classes match the orbits of the Lie group $\mathcal{G}_{\mathbf{k}}$ of tuples (A_1, \dots, A_{d-1}) of invertible matrices acting on $\mathcal{W}_{\mathbf{k}}^*$ through (9.33). One can then apply a common procedure in differential geometry and first establish that the quotient space $\mathcal{W}_{\mathbf{k}}^*/\mathcal{G}_{\mathbf{k}}$ possesses a smooth manifold structure such that the quotient map $\mathcal{W}_{\mathbf{k}}^* \rightarrow \mathcal{W}_{\mathbf{k}}^*/\mathcal{G}_{\mathbf{k}}$ is a submersion. As a second step, one shows that the parametrization $\mathcal{W}_{\mathbf{k}}^*/\mathcal{G}_{\mathbf{k}} \rightarrow \mathcal{M}_{\mathbf{k}}$ by the quotient manifold is an injective immersion and a homeomorphism in the topology of the ambient space $\mathbb{R}^{n_1 \times \dots \times n_d}$. It then follows from standard results

(see, e.g., [66, Prop. 8.3]), that $\mathcal{M}_{\mathbf{k}}$ is an embedded submanifold of $\mathbb{R}^{n_1 \times \dots \times n_d}$ and its dimension is

$$\dim(\mathcal{M}_{\mathbf{k}}) = \dim(\mathcal{W}_{\mathbf{k}}^*) - \dim(\mathcal{G}_{\mathbf{k}}) = 1 + \sum_{\mu=1}^d r_{\mu-1} n_{\mu} r_{\mu} - r_{\mu}^2. \quad (9.34)$$

The details of this construction can be found in [108].

9.3.4 Tangent Space and Retraction

In view of the practical geometric methods on the manifold $\mathcal{M}_{\mathbf{k}}$ to be described later, we now consider the efficient representation of tangent vectors and the computation of retractions. These are quite analogous to the matrix case. First of all, using, e.g., the compact notation (9.21) for the multilinear and surjective parametrization $(G_1, \dots, G_d) \mapsto X$ of the TT manifold $\mathcal{W}_{\mathbf{k}}^*$, it is clear that the tangent space $T_X \mathcal{M}_{\mathbf{k}}$ at a point $X \in \mathcal{M}_{\mathbf{k}}$ with TT-cores $(G_1, \dots, G_d) \in \mathcal{W}_{\mathbf{k}}^*$ (see Sect. 9.3.3) consists of all tensors ξ of the form

$$\xi(i_1, \dots, i_d) = \sum_{\mu=1}^d G_1(i_1) \cdots G_{\mu-1}(i_{\mu-1}) \dot{G}_{\mu}(i_{\mu}) G_{\mu+1}(i_{\mu+1}) \cdots G_d(i_d), \quad (9.35)$$

where the cores \dot{G}_{μ} at position μ can be chosen freely. In view of (9.34), this representation has too many degrees of freedom, even when fixing the TT decomposition G_1, \dots, G_d of X , but this redundancy can be removed by gauging conditions.

A very reasonable way to do this is the following [56, 103]. We assume that the cores U_1, \dots, U_{d-1} and V_2, \dots, V_d for the orthogonal decompositions (9.24)–(9.25) are available. Then, since the \dot{G}_{μ} in (9.35) are entirely free, we do not lose generality by orthogonalizing every term of the sum around \dot{G}_{μ} :

$$\xi(i_1, \dots, i_d) = \sum_{\mu=1}^d U_1(i_1) \cdots U_{\mu-1}(i_{\mu-1}) \dot{G}_{\mu}(i_{\mu}) V_{\mu+1}(i_{\mu+1}) \cdots V_d(i_d). \quad (9.36)$$

We now can add the gauging conditions

$$(U_{\mu}^{(2)})^T \dot{G}_{\mu}^{(2)} = 0, \quad \mu = 1, \dots, d-1, \quad (9.37)$$

which remove r_{μ}^2 degrees of freedom from each of the cores $\dot{G}_1, \dots, \dot{G}_{d-1}$. The last core \dot{G}_d is not constrained.

What this representation of tangent vectors achieves is that all d terms in (9.36) now reside in mutually orthogonal subspaces T_1, \dots, T_d . In other words, the tangent space $T_X \mathcal{M}_{\mathbf{k}}$ is orthogonally decomposed:

$$T_X \mathcal{M}_{\mathbf{k}} = T_1 \oplus \dots \oplus T_d.$$

This allows to write the orthogonal projection onto $T_X \mathcal{M}_{\mathbf{k}}$ as a sum of orthogonal projections onto the spaces T_1, \dots, T_d . To derive these projections, consider first the operators that realize the orthogonal projection onto the row and column space of the unfoldings $X^{(\mu)}$. They read

$$\mathcal{P}_{\leq \mu}(Z) = \text{Ten}_{\mu}(U_{\leq \mu} U_{\leq \mu}^T Z^{(\mu)}) \quad \text{and} \quad \mathcal{P}_{\geq \mu+1}(Z) = \text{Ten}_{\mu}(Z^{(\mu)} V_{\geq \mu+1} V_{\geq \mu+1}^T), \tag{9.38}$$

where Ten_{μ} denotes the inverse operation of the μ th unfolding so that $\mathcal{P}_{\leq \mu}$ and $\mathcal{P}_{\geq \mu+1}$ are in fact orthogonal projectors in the space $\mathbb{R}^{n_1 \times \dots \times n_d}$. Note that $\mathcal{P}_{\leq \mu}$ and $\mathcal{P}_{\geq \nu}$ commute when $\mu < \nu$. Furthermore, $\mathcal{P}_{\leq \mu} \mathcal{P}_{\leq \nu} = \mathcal{P}_{\leq \nu}$ and $\mathcal{P}_{\geq \nu} \mathcal{P}_{\geq \mu} = \mathcal{P}_{\geq \mu}$ if $\mu < \nu$.

By inspecting the different terms in (9.36) and taking the gauging (9.37) into account, it is not so difficult to verify that the projection on T_1 is given by

$$Z \mapsto (I - \mathcal{P}_{\leq 1}) \mathcal{P}_{\geq 2} Z,$$

the projection on T_2 is given by

$$Z \mapsto \mathcal{P}_{\leq 1} (I - \mathcal{P}_{\leq 2}) \mathcal{P}_{\geq 3} Z = (\mathcal{P}_{\leq 1} - \mathcal{P}_{\leq 2}) \mathcal{P}_{\geq 3} Z$$

and so forth. Setting $\mathcal{P}_{\leq 0} = \mathcal{P}_{\geq d+1} = I$ (identity) for convenience, the overall projector \mathcal{P}_X onto the tangent space $T_X \mathcal{M}_{\mathbf{k}}$ is thus given in one of the two following forms [72]:

$$\begin{aligned} \mathcal{P}_X &= \sum_{\mu=1}^{d-1} (\mathcal{P}_{\leq \mu-1} - \mathcal{P}_{\leq \mu}) \mathcal{P}_{\geq \mu+1} + \mathcal{P}_{\leq d-1} \\ &= \mathcal{P}_{\geq 2} + \sum_{\mu=2}^d \mathcal{P}_{\leq \mu-1} (\mathcal{P}_{\geq \mu+1} - \mathcal{P}_{\mu}). \end{aligned} \tag{9.39}$$

The formulas (9.39) for the projector on the tangent space are conceptually insightful but still extrinsic. An efficient implementation of this projection for actually getting the gauged components \dot{G}_{μ} that represent the resulting tangent vector is possible if Z is itself a TT tensor of small ranks or a very sparse tensor. For example, due to the partial TT structure of projectors (9.38), when computing $\mathcal{P}_{\leq \mu+1} Z$, the partial result from $\mathcal{P}_{\leq \mu} Z$ can be reused and so on. The full details are cumbersome to explain so we do not present them here and refer to [73, §7] and [103, §4].

It is also interesting to note that the tangent space $T_X \mathcal{M}_{\mathbf{k}}$ itself contains only tensors of TT rank at most $2\mathbf{k}$. This is due to the structure (9.35) of tangent vectors as sums of TT decompositions that vary in a single core each [50]. Since X itself is in $T_X \mathcal{M}_{\mathbf{k}}$, we directly write the TT decomposition of $X + \xi$, since this will be the tensors that need to be retracted in optimization methods. In terms of the left- and right-orthogonal cores U_1, \dots, U_{d-1} and V_2, \dots, V_d from (9.25) we have [103]

$$(X + \xi)(i_1, \dots, i_d) = W_1(i_1)W_2(i_2) \cdots W_{d-1}(i_{d-1})W_d(i_d), \tag{9.40}$$

with the cores

$$W_1(i_1) = \begin{bmatrix} U_1(i_1) & \dot{G}_1(i_1) \end{bmatrix}, \quad W_\mu(i_\mu) = \begin{bmatrix} U_\mu(i_\mu) & \dot{G}_\mu(i_\mu) \\ 0 & V_\mu(i_\mu) \end{bmatrix}$$

for $\mu = 2, \dots, d - 1$, and

$$W_d(i_d) = \begin{bmatrix} S_d V_d(i_d) + \dot{G}_d(i_d) \\ V_d(i_d) \end{bmatrix},$$

where S_d is the matrix from the d -orthogonal decomposition (9.24) of X . The formula (9.40) is the TT analog to (9.14).

Finally we mention that since $\mathcal{M}_{\mathbf{k}}$ is a smooth manifold, the best approximation of $X + \xi$ would be in principle a feasible retraction from the tangent space to the manifold. It is, however, computationally not available. The TT-SVD algorithm applied to $X + \xi$ with target ranks \mathbf{k} is a valid surrogate, which due to the TT representation (9.40) of tangent vectors is efficiently applicable. As discussed in Sect. 9.3.2 the TT-SVD procedure is essentially a composition of nonlinear projections on low-rank matrix manifolds, which are locally smooth around a given $X \in \mathcal{M}_{\mathbf{k}}$. This provides the necessary smoothness properties of the TT-SVD algorithm when viewed as a projection on $\mathcal{M}_{\mathbf{k}}$. On the other hand, the quasi-optimality of this projection as established in (9.30) implies the retraction property (9.13); see [103] for the details.

9.3.5 Elementary Operations and TT Matrix Format

Provided that the ranks are small enough, the TT representation introduced above allows to store very high-dimensional tensors in practice and to access each entry individually by computing the matrix product (9.21). Furthermore, it is possible to efficiently perform certain linear algebra operations. For instance the sum of two TT tensors X and \hat{X} with TT cores G_1, \dots, G_d and $\hat{G}_1, \dots, \hat{G}_d$ has the matrix product representation

$$(X + \hat{X})(i_1, \dots, i_d) = [G_1(i_1) \hat{G}_1(i_1)] \begin{bmatrix} G_2(i_2) & 0 \\ 0 & \hat{G}_2(i_2) \end{bmatrix} \cdots \begin{bmatrix} G_{d-1}(i_{d-1}) & 0 \\ 0 & \hat{G}_{d-1}(i_{d-1}) \end{bmatrix} \begin{bmatrix} G_d(i_d) \\ \hat{G}_d(i_d) \end{bmatrix}.$$

Hence the core tensors are simply augmented, and no addition at all is required when implementing this operation. Note that this shows that the TT rank of $X + \hat{X}$ is bounded by the (entry-wise) sum of TT ranks of X and \hat{X} .

As another example, the Frobenius inner product of X and \hat{X} can be implemented by performing the nested summation in

$$\langle X, \hat{X} \rangle_F = \sum_{i_1=1}^{n_1} \cdots \sum_{i_d=1}^{n_d} G_1(i_1) \cdots G_d(i_d) \hat{G}_d(i_d)^T \cdots \hat{G}_1(i_1)^T$$

sequentially: first, the matrix

$$Z_d = \sum_{i_d=1}^{n_d} G_d(i_d) \hat{G}_d(i_d)^T$$

is computed, then

$$Z_{d-1} = \sum_{i_{d-1}=1}^{n_{d-1}} G_{d-1}(i_{d-1}) Z_d \hat{G}_{d-1}(i_{d-1})^T$$

and so on. These computations only involve matrix products and the final result Z_1 will be the desired inner product. The computational complexity for computing inner products is hence $O(dnr^3)$ with $n = \max n_\mu$ and $r = \max\{r_\mu, \hat{r}_\mu\}$, where \mathbf{r} and $\hat{\mathbf{r}}$ are the TT-ranks of X and \hat{X} , respectively. As a special case, the Frobenius norm of a TT tensor can be computed.

Obviously, these elementary operations are crucial for applying methods from numerical linear algebra and optimization. However, in many applications the most important operation is the computation of the ‘matrix-vector-product’, that is, in our case the action of a given linear operator \mathcal{A} on a tensor X . In order to use low-rank techniques like Riemannian optimization it is mandatory that the given operator \mathcal{A} can be applied efficiently. In some applications, sparsity of \mathcal{A} makes this possible. More naturally, most low-rank formats for tensors come with a corresponding low-rank format for linear operators acting on such tensors that enable their efficient application. For the TT format, the corresponding operator format is called the *TT matrix format* [86] or *matrix product operator* (MPO) format [115].

A linear map $\mathcal{A}: \mathbb{R}^{n_1 \times \cdots \times n_d} \rightarrow \mathbb{R}^{n_1 \times \cdots \times n_d}$ can be identified with an $(n_1 \cdots n_d) \times (n_1 \cdots n_d)$ matrix with entries $[\mathcal{A}(i_1, \dots, i_d; j_1, \dots, j_d)]$, where both the rows and columns are indexed with multi-indices. The operator \mathcal{A} is then said to be in the TT matrix format with *TT matrix ranks* (R_1, \dots, R_{d-1}) if its entries can be written as

$$\mathcal{A}(i_1, \dots, i_d; j_1, \dots, j_d) = O_1(i_1, j_1)O_2(i_2, j_2) \cdots O_d(i_d, j_d),$$

where $O_\mu(i_\mu, j_\mu)$ are matrices of size $R_{\mu-1} \times R_\mu$ ($R_0 = R_d = 1$). Clearly, the TT matrix format becomes the usual TT format when treating \mathcal{A} as an $n_1^2 \times \cdots \times n_d^2$ tensor.

Note that if \mathcal{A} is an operator on matrices, that is, in the case $d = 2$, $O_1(i_\mu, j_\mu)$ and $O_2(i_\mu, j_\mu)$ are just vectors of length $R_1 = R$, and the formula can be written as

$$\mathcal{A}(i_1, i_2; j_1, j_2) = \sum_{\ell=1}^R O_{1,\ell}(i_1, j_1)O_{2,\ell}(i_2, j_2).$$

In other words, such an operator \mathcal{A} is a sum

$$\mathcal{A} = \sum_{\ell=1}^R A_\ell \otimes B_\ell$$

of Kronecker products of matrices $[A_\ell(i, j)] = [O_{1,\ell}(i, j)]$ and $[B_\ell(i, j)] = [O_{2,\ell}(i, j)]$.

An operator in the TT matrix format can be efficiently applied to a TT tensor, yielding a result in the TT format again. Indeed, let $Y = \mathcal{A}(X)$, then a TT decomposition of Y can be found using the properties of the Kronecker product \otimes of matrices [86]:

$$\begin{aligned} Y(i_1, \dots, i_d) &= \sum_{j_1=1}^{n_1} \cdots \sum_{j_d=1}^{n_d} \mathcal{A}(i_1, \dots, i_d; j_1, \dots, j_d)X(j_1, \dots, j_d) \\ &= \sum_{j_1=1}^{n_1} \cdots \sum_{j_d=1}^{n_d} (O_1(i_1, j_1) \cdots O_d(i_d, j_d)) \otimes (G_1(j_1) \cdots G_d(j_d)) \\ &= \sum_{j_1=1}^{n_1} \cdots \sum_{j_d=1}^{n_d} (O_1(i_1, j_1) \otimes G_1(j_1)) \cdots (O_d(i_d, j_d) \otimes G_d(j_d)) \\ &= \hat{G}_1(i_1) \cdots \hat{G}_d(i_d) \end{aligned}$$

with resulting TT cores

$$\hat{G}_\mu(i_\mu) = \sum_{j_\mu=1}^{n_\mu} O_\mu(i_\mu, j_\mu) \otimes G_\mu(j_\mu), \quad \mu = 1, \dots, d.$$

Forming all these cores has a complexity of $O(dnr^2R^2)$, where $R = \max R_\mu$.

Note that $G_\mu(i_\mu)$ is a matrix of size $r_{\mu-1}R_{\mu-1} \times r_\mu R_\mu$ so the TT ranks of \mathcal{A} and X are multiplied when applying \mathcal{A} to X . In algorithms where this operation is

performed several times it therefore can become necessary to apply the TT-SVD procedure to the result as a post-processing step for reducing ranks again. This is akin to rounding in floating point arithmetic and is therefore also called TT-rounding.

9.4 Optimization Problems

As we have explained above, the sets of matrices of fixed rank k and tensors of fixed TT rank \mathbf{k} are smooth submanifolds $\mathcal{M}_k \subset \mathbb{R}^{m \times n}$ and $\mathcal{M}_{\mathbf{k}} \subset \mathbb{R}^{n_1 \times \dots \times n_d}$, respectively. In this section we will see how to efficiently exploit these smooth structures in optimization problems.

Here and in the following \mathbb{V} denotes a finite dimensional real vector space, that depending on the context, can be just \mathbb{R}^N , a space $\mathbb{R}^{m \times n}$ of matrices, or a space $\mathbb{R}^{n_1 \times \dots \times n_d}$ of tensors.

9.4.1 Riemannian Optimization

We start with a relatively general introduction to local optimization methods on smooth manifolds; see [2] for a broader but still self-contained treatment of this topic.

Let \mathcal{M} be a smooth submanifold in \mathbb{V} , like \mathcal{M}_k or $\mathcal{M}_{\mathbf{k}}$. Since $\mathcal{M} \subset \mathbb{V}$, we can represent a point X on \mathcal{M} as an element of \mathbb{V} . We can do the same for its tangent vectors $\xi \in T_X \mathcal{M}$ since $T_X \mathcal{M} \subset T_X \mathbb{V} \simeq \mathbb{V}$. This allows us to restrict any smoothly varying inner product on \mathbb{V} to $T_X \mathcal{M}$ and obtain a Riemannian metric $(\cdot, \cdot)_X$ on \mathcal{M} . For simplicity, we choose the Euclidean metric:

$$(\xi, \eta)_X = \xi^T \eta, \quad \xi, \eta \in T_X \mathcal{M} \subset \mathbb{V}.$$

Consider now a smooth objective function $f: \mathbb{V} \rightarrow \mathbb{R}$. If we restrict its domain to \mathcal{M} , we obtain an optimization problem on a Riemannian manifold:

$$\min f(X) \quad \text{s.t.} \quad X \in \mathcal{M}. \quad (9.41)$$

The aim of a Riemannian optimization method is to generate iterates X_1, X_2, \dots that remain on \mathcal{M} and converge to a (local) minimum of f constrained to \mathcal{M} ; see Fig. 9.2. It uses only local knowledge of f , like first and second-order derivatives. It thus belongs to the family of feasible methods for constrained optimization, which is a very useful property in our setting since general tensors or matrices in \mathbb{V} with arbitrary rank might otherwise be too large to store. A distinctive difference with other methods for constrained optimization is that a Riemannian optimization method has a detailed geometric picture of the constraint set \mathcal{M} at its disposal.

Fig. 9.2 A Riemannian optimization method generates iterates X_ℓ starting from X_1 to minimize f on a manifold \mathcal{M} . The thin gray lines are level sets of f and X_* is a (local) minimum of f on \mathcal{M}

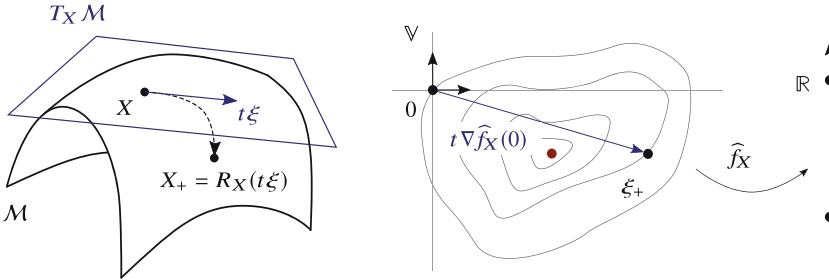
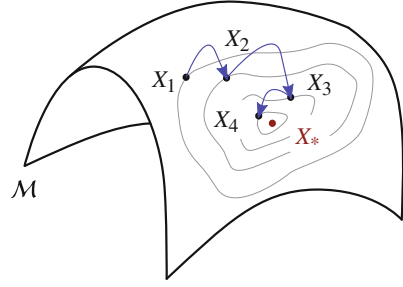


Fig. 9.3 One step of a typical Riemannian optimization method with step direction ξ on the submanifold (left). Example of one step of steepest descent on the pullback (right)

In its most basic form, a Riemannian optimization method is the update formula

$$X_+ = R_X(t \xi), \tag{9.42}$$

that is then repeated after replacing X by X_+ . The formula (9.42) is defined by the following ‘ingredients’; see also the left panel of Fig. 9.3.

1. The *search direction* $\xi \in T_X \mathcal{M}$ that indicates the direction of the update. Similar as in Euclidean unconstrained optimization, the search direction can be obtained from first-order (gradient) or second-order (Hessian) information.⁶ Generally, f will locally decrease in the direction of ξ , that is, the directional derivative satisfies $f'(X) \xi < 0$.
2. As explained in Sect. 9.2.4, the *retraction* $R_X: T_X \mathcal{M} \rightarrow \mathcal{M}$ is a smooth map that replaces the usual update $X + t \xi$ from Euclidean space to the manifold setting. Running over t , we thus replace a straight ray with a curve that (locally) lies on \mathcal{M} by construction. By the retraction property (9.13), the curve is rigid at $t = 0$, which means that $R_X(0) = X$ and $\frac{d}{dt} R(t\xi)|_{t=0} = \xi$ for all $\xi \in T_X \mathcal{M}$.
3. The *step size* $t > 0$ is usually chosen to guarantee sufficient decrease of f in X_+ , although non-monotone strategies also exist. Given ξ , the step size is typically

⁶We stick here to more standard smooth optimization on purpose but also nonsmooth and stochastic methods are possible for Riemannian manifolds; see [38, 47, 48, 95].

found by line search strategies like backtracking, whereas an *exact* line search would provide a global minimum along direction ξ , if it exists. As an alternative one can use the trust-region mechanism to generate $t\xi$.

To explain the possible search directions ξ at a point $X \in \mathcal{M}$, we take a slight detour and consider the *pullback* of f at X :

$$\widehat{f}_X = f \circ R_X: T_X \mathcal{M} \rightarrow \mathcal{M}.$$

Since \widehat{f}_X is defined on the linear subspace $T_X \mathcal{M}$, we can for example minimize it by the standard steepest descent method; see the right panel of Fig. 9.3. Observe that rigidity of R_X implies $\widehat{f}_X(0) = f(X)$. Hence, the starting guess is the zero tangent vector, which will get updated as

$$\xi_+ = 0 - \beta^\ell \beta_0 \nabla \widehat{f}_X(0)$$

and *Armijo backtracking* determines the smallest $\ell = 0, 1, \dots$ such that

$$\widehat{f}_X(\xi_+) \leq \widehat{f}_X(0) - c \beta^\ell \beta_0 \|\nabla \widehat{f}_X(0)\|_F^2. \quad (9.43)$$

Here, $\beta = 1/2$, $\beta_0 = 1$, and $c = 0.99$ are standard choices. We could keep on iterating, but the crucial point is that in Riemannian optimization, we perform such a step only once, and then redefine the pullback function for $X_+ = R_X(\xi_+)$ before repeating the procedure.

Formally, the iteration just described is clearly of the form as (9.42), but it is much more fruitful to regard this procedure from a geometric point of view. To this end, observe that rigidity of R_X also implies

$$\widehat{f}'_X(0) \xi = f'(X) R'_X(0) \xi = f'(X) \xi \quad \text{for all } \xi \in T_X \mathcal{M}.$$

With $\mathcal{P}_X: \mathbb{V} \rightarrow T_X \mathcal{M}$ the orthogonal projection, we thus obtain

$$(\nabla \widehat{f}_X(0), \xi)_F = (\nabla f(X), \mathcal{P}_X(\xi))_F = (\mathcal{P}_X(\nabla f(X)), \xi)_F. \quad (9.44)$$

These identities allow us to define the *Riemannian gradient* of f at X to \mathcal{M} simply as the tangent vector $\mathcal{P}_X(\nabla f(X))$. This vector is conveniently also a direction of steepest ascent among all tangent vectors at X with the same length. We can thus define the *Riemannian steepest descent method* as

$$X_+ = R_X(-t \mathcal{P}_X(\nabla f(X))), \quad \text{with } t = \beta^\ell \beta_0. \quad (9.45)$$

Here, Armijo backtracking picks again the smallest ℓ (since $0 < \beta < 1$) such that

$$f(R_X(X_+)) \leq f(X) - c \beta^\ell \beta_0 \|\mathcal{P}_X \nabla f(X)\|_F^2.$$

Observe that we have arrived at the same iteration as above but instead of using a pullback we derived it directly from geometric concepts, where we have benefited from choosing the Euclidean metric on $T_X \mathcal{M}$ for obtaining the simple formula (9.44) for the Riemannian gradient. Using the notion of second-order retractions, one can in this way derive the *Riemannian Newton method* either using the Riemannian Hessian with pullbacks or directly with the Riemannian connection. We refer to [2] for details, where also trust-region strategies are discussed.

The ‘recipe’ above leaves a lot of freedom, which can be used to our advantage to choose computational efficient components that work well in practice. Below we will focus on approaches that are ‘geometric versions’ of classical, non-Riemannian algorithms, yet can be implemented efficiently on a manifold so that they become competitive.

9.4.2 Linear Systems

We now explain how Riemannian optimization can be used to solve very large linear systems. Given a linear operator $\mathcal{L}: \mathbb{V} \rightarrow \mathbb{V}$ and a ‘right-hand side’ $B \in \mathbb{V}$, the aim is to calculate any X_{ex} that satisfies the equation

$$\mathcal{L}(X_{\text{ex}}) = B.$$

Since our strategy is optimization, observe that X_{ex} can also be found as a global minimizer of the residual objective function

$$f_{LS}(X) = (\mathcal{L}(X) - B, \mathcal{L}(X) - B)_F = \|\mathcal{L}(X) - B\|_F^2.$$

If in addition \mathcal{L} is symmetric and positive semi-definite on \mathbb{V} , the same is true for the energy norm function

$$\begin{aligned} f_{\mathcal{L}}(X) &= (X, \mathcal{L}(X))_F - 2(X, B)_F \\ &= (X - X_{\text{ex}}, \mathcal{L}(X - X_{\text{ex}}))_F - (X_{\text{ex}}, \mathcal{L}(X_{\text{ex}}))_F. \end{aligned}$$

The second identity shows that $f_{\mathcal{L}}(X)$ is indeed, up to a constant, the square of the error in the induced \mathcal{L} -(semi)norm. In the following, we will assume that \mathcal{L} is positive semi-definite and focus only on $f = f_{\mathcal{L}}$ since it leads to better conditioned problems compared to f_{LS} .

When X_{ex} is a large matrix or tensor, we want to approximate it by a low-rank matrix or tensor. Since we do not know X_{ex} we cannot use the quasi-best truncation procedures as explained in Sect. 9.3. Instead, we minimize the restriction of $f = f_{\mathcal{L}}$ onto an approximation manifold $\mathcal{M} = \mathcal{M}_k$ or $\mathcal{M} = \mathcal{M}_k$:

$$\min f(X) \quad \text{s.t.} \quad X \in \mathcal{M}.$$

This is exactly a problem of the form (9.41) and we can, for example, attempt to solve it with the Riemannian steepest descent algorithm. With $X \in \mathcal{M}_k$ and the definition of $f_{\mathcal{L}}$, this iteration reads

$$X_+ = R_X(-t \mathcal{P}_X(\mathcal{L}(X) - B)). \quad (9.46)$$

When dealing with ill-conditioned problems, as they occur frequently with discretized PDEs, it is advisable to include some preconditioning. In the Riemannian context, one way of doing this is by modifying (9.46) to

$$X_+ = R_X(-t \mathcal{P}_X(Q(\mathcal{L}(X) - B))), \quad (9.47)$$

where $Q: \mathbb{V} \rightarrow \mathbb{V}$ is a suitable preconditioner for \mathcal{L} . This iteration is called *truncated Riemannian preconditioned Richardson iteration* in [65] since it resembles a classical Richardson iteration.

9.4.3 Computational Cost

Let us comment which parts of (9.46) are typically the most expensive. Since the retraction operates on a tangent vector, it is cheap both for matrices and tensors in TT format as long as their ranks are moderate; see Sect. 9.3. The remaining potentially expensive steps are therefore the application of the projector \mathcal{P}_X and the computation of the step size t .

Let $Z = \mathcal{L}(X) - B$ be the residual. Recall that the projected tangent vector $\xi = \mathcal{P}_X(Z)$ will be computed using (9.12) for matrices and (9.36)–(9.37) for TT tensors. As briefly mentioned before, these formulas are essentially many (unfolded) matrix multiplications that can efficiently be computed if Z is a *sparse* or *low rank* matrix/tensor.

Sparsity occurs for example in the matrix and tensor completion problems (see Sect. 9.6 later) where \mathcal{L} is the orthogonal projector \mathcal{P}_{Ω} onto a sampling set $\Omega \subset \{1, \dots, n_1\} \times \dots \times \{1, \dots, n_d\}$ of known entries of an otherwise unknown matrix/tensor $X_{\text{ex}} \in \mathbb{V}$. The matrix/tensor B in this problem is then the sparse matrix/tensor containing the known entries of X_{ex} . Then if, for example, $X = USV^T \in \mathcal{M}_k$ is a matrix in SVD-like format, the residual $Z = \mathcal{P}_{\Omega}(X) - B$ is also a sparse matrix whose entries are computed as

$$Z(i_1, i_2) = \begin{cases} \sum_{\ell=1}^r U(i_1, \ell) S(\ell, \ell) V(i_2, \ell) - B(i_1, i_2) & \text{if } (i_1, i_2) \in \Omega, \\ 0 & \text{otherwise.} \end{cases}$$

Hence the computation of $\mathcal{P}_X(Z)$ now requires two sparse matrix multiplications ZU and $Z^T V$; see [112]. For tensor completion, a little bit more care is needed but

an efficient implementation for applying the tangent space projector exists; see [103, §4.2]. In all cases, the computation becomes cheaper the sparser Z is.

If on the other hand \mathcal{L} is a low-rank TT matrix operator as explained in Sect. 9.3.5, and B is a low-rank TT tensor, then $Z = \mathcal{L}(X) - B$ will be also of low-rank since $X \in \mathcal{M}_k$. This makes the tangent space projection $\mathcal{P}_X(Z)$ efficiently applicable afterwards as explained before. Operators with TT matrix structure are the most typical situation when TT tensors are used for parametric PDEs and for the Schrödinger equation; see again Sect. 9.6 later.

Regarding the computation of the step size t , we can approximate an exact line search method by minimizing the first-order approximation

$$g(t) = f(X - t\xi) \approx f(R_X(-t\xi)).$$

For quadratic functions f , the function $g(t)$ is a quadratic polynomial in t and can thus be exactly minimized. For instance, with $f_{\mathcal{L}}$ it satisfies

$$g(t) = (\xi, \mathcal{L}(\xi))_F t^2 - 2(\xi, \mathcal{L}(X) - B)_F t + \text{constant}.$$

Recall that, by (9.40), the matrix or TT rank of a tangent vector ξ is bounded by two times that of X . Hence, in the same situation as for \mathcal{L} above, these inner products can be computed very efficiently. It has been observed in [112] that with this initialization of the step size almost no extra backtracking is needed.

9.4.4 Difference to Iterative Thresholding Methods

A popular algorithm for solving optimization problems with low-rank constraints, like matrix completion [49] and linear tensor systems [8, 54], is *iterative hard thresholding* (IHT).⁷ It is an iteration of the form

$$X_+ = \mathcal{P}_{\mathcal{M}}(X - t \nabla f(X)),$$

where $\mathcal{P}_{\mathcal{M}}: \mathbb{V} \rightarrow \mathcal{M}$ denotes the (quasi) projection on the set \mathcal{M} , like the truncated SVD for low-rank matrices and TT-SVD for tensors as explained in Sects. 9.2.1 and 9.3.2. Variations of this idea also include alternating projection schemes like in [101]. Figure 9.4 compares IHT to Riemannian steepest descent. The main difference between the two methods is the extra tangent space projection \mathcal{P}_X of the negative gradient $-\nabla f(X)$ for the Riemannian version. Thanks to this projection, the truncated SVD in the Riemannian case has to be applied to a tangent vector which can be implemented cheaply with direct linear algebra and is thus very reliable, as explained in Sects. 9.2.4 and 9.3.4. In IHT on the other hand, the

⁷Also called *singular value projection* and *truncated Richardson iteration*.

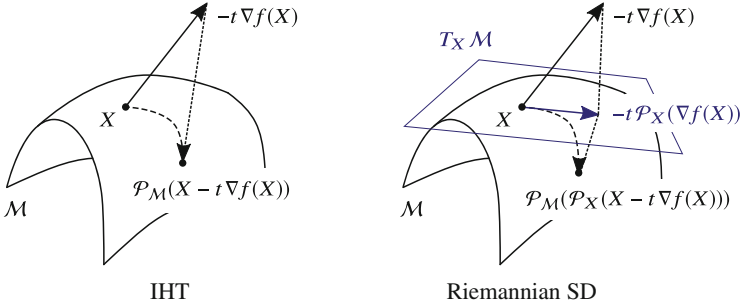


Fig. 9.4 Iterative hard thresholding (IHT) and Riemannian steepest descent (SD) for fixed-rank matrices and tensors

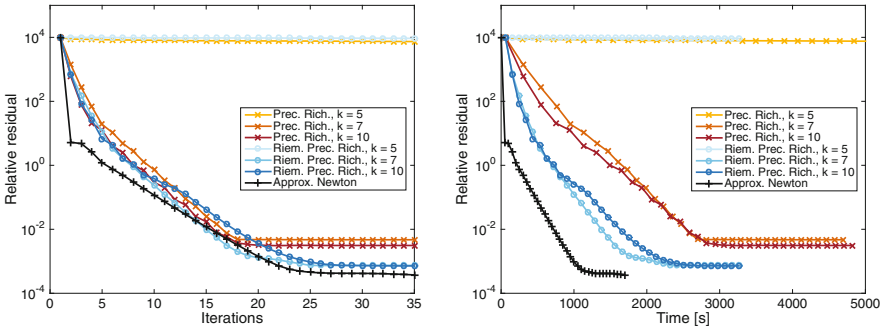


Fig. 9.5 Convergence of the Riemannian and non-Riemannian versions of preconditioned Richardson iteration. The approximation quality of the preconditioner is proportional to the k value. (We do not explain the other Riemannian method “approx. Newton”.) Picture taken from [65]. Copyright 2016 Society for Industrial and Applied Mathematics. Reprinted with permission. All rights reserved

truncated SVD is applied to a generally unstructured search direction and needs to be implemented with sparse or randomized linear algebra, which are typically less reliable and more expensive.

This difference becomes even more pronounced with preconditioning for linear systems $\mathcal{L}(X) = B$ as in (9.47). As approximate inverse of \mathcal{L} , the operator Q there has typically high TT matrix rank and so the additional tangent space projector in (9.47) is very beneficial compared to the seemingly more simpler *truncated preconditioned Richardson method*

$$X_+ = \mathcal{P}_M(X - tQ(\mathcal{L}(X) - B)).$$

The numerical experiments from [65] confirm this behavior. For example, in Fig. 9.5 we see the convergence history when solving a Laplace-type equation with Newton potential in the low-rank Tucker format, which has not been discussed, but illustrates the same issue. Since the Newton potential is approximated by a rank 10 Tucker

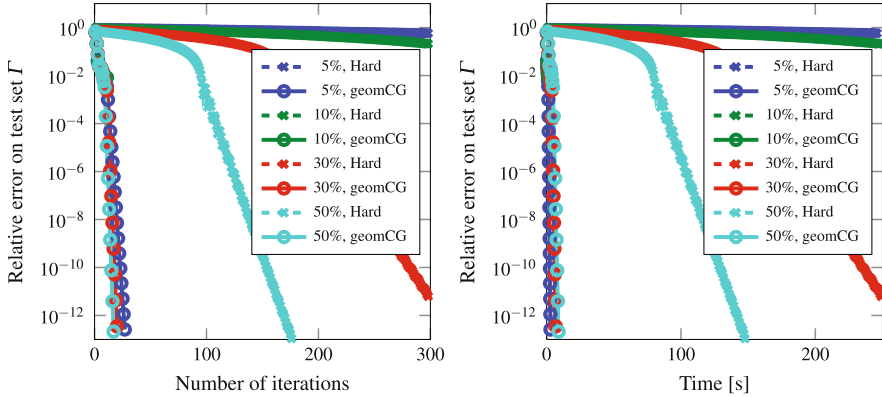


Fig. 9.6 Relative testing error in function of number of iterations (left) and time (right). For an IHT algorithm (denoted “Hard”) and a Riemannian method (denoted by “geomCG”) for different sampling sizes when solving the tensor completion problem. Picture taken from [64]

matrix, applying QL greatly increases the rank of the argument. Thanks to the tangent space projections, the time per iteration is reduced significantly and there is virtually no change in the number of iterations needed.

There is another benefit of Riemannian algorithms over more standard rank truncated schemes. Thanks to the global smoothness of the fixed-rank manifolds \mathcal{M} , it is relatively straightforward to accelerate manifold algorithms using non-linear CG or BFGS, and perform efficient line search. For example, Fig. 9.6 compares the Riemannian non-linear CG algorithm from [64] to a specific IHT algorithm based on nuclear norm relaxation from [101] for the low-rank tensor completion problem as explained in Sect. 9.6.3. We can see that the Riemannian algorithm takes less iterations and less time. While this example is again for fixed-rank Tucker tensors, the same conclusion is also valid for fixed-rank matrices and TT tensors; see, e.g., [112, Fig. 5.1].

9.4.5 Convergence

Theoretical results for Riemannian optimization parallel closely the results from Euclidean unconstrained optimization. In particular, with standard line search or trust-region techniques, limit points are guaranteed to be critical points, and additional Hessian information can enforce attraction to local minimal points; see [2]. For example, when the initial point X_1 is sufficiently close to a strict local minimizer X_* of f on \mathcal{M} , Riemannian gradient descent will converge exponentially fast. Specifically, if the Riemannian Hessian of f at X_* has all positive eigenvalues $\lambda_p \geq \dots \geq \lambda_1 > 0$, then the iterates X_ℓ with exact line search satisfy the following asymptotic Q-linear convergence rate [74]:

$$\lim_{\ell \rightarrow \infty} \frac{\|X_{\ell+1} - X_*\|_F}{\|X_\ell - X_*\|_F} = \frac{\lambda_p - \lambda_1}{\lambda_p + \lambda_1} \leq 1 - \kappa, \quad \text{with } \kappa = \frac{\lambda_1}{\lambda_p}.$$

With more practical line searches, like those that ensure the Armijo condition (9.43), this rate deteriorates but remains $1 - O(\kappa)$; see [2]. As in the Euclidean non-convex case, non-asymptotic results that are valid for arbitrary X_1 can only guarantee algebraic rates; see [12]. If however X_1 is in a region where f is locally convex, then also fast exponential convergence is guaranteed; see [107]. Results of this kind but specific to matrix completion are available in [117].

For particular problems, one can show that gradient schemes converge to the global minimum when started at any X_1 . The main idea is that, while these problems are not convex, their optimization landscape is still favorable for gradient schemes in the sense that all critical points are either strict saddle points or close to a global minimum. Strict saddles are characterized as having directions of sufficient negative curvature so that they push away the iterates of a gradient scheme that might be attracted to such a saddle [76]. This property has been established in detail for matrix sensing with RIP (restricted isometry property) operators, which are essentially very well-conditioned when applied to low-rank matrices. Most of the results are formulated for particular non-Riemannian algorithms (see, e.g., [90]), but landscape properties can be directly applied to Riemannian algorithms as well; see [18, 110]. As far as we know, such landscape results have not been generalized to TT tensors but related work on completion exists [93].

9.4.6 Eigenvalue Problems

Another class of optimization problems arises when computing extremal eigenvalues of Hermitian operators. This is arguably the most important application of low-rank tensors in theoretical physics since it includes the problem of computing ground-states (eigenvectors of minimal eigenvalues) of the Schrödinger equation.

The main idea is similar to the previous section. Suppose we want to compute an eigenvector $X \in \mathbb{V}$ of a minimal eigenvalue of the Hermitian linear operator $\mathcal{H}: \mathbb{V} \rightarrow \mathbb{V}$. Then, instead of minimizing the Rayleigh function on \mathbb{V} , we restrict the optimization space to an approximation manifold:

$$\min \rho(X) = \frac{(X, \mathcal{H}(X))_F}{(X, X)_F} \quad \text{s.t.} \quad X \in \mathcal{M}.$$

Since f is homogeneous in X , the normalization $(X, X)_F = 1$ can also be imposed as a constraint:

$$\min \tilde{\rho}(X) = (X, \mathcal{H}(X))_F \quad \text{s.t.} \quad X \in \tilde{\mathcal{M}} = \mathcal{M} \cap \{X: (X, X)_F = 1\}.$$

This intersection is transversal in cases when \mathcal{M} is a manifold of low-rank matrices or tensors, so $\tilde{\mathcal{M}}$ is again a Riemannian submanifold of \mathbb{V} with a geometry very similar to that of \mathcal{M} ; see [91] for details on the matrix case. One can now proceed and apply Riemannian optimization to either problem formulation.

Standard algorithms for eigenvalue problems typically do not use pure gradient schemes. Thanks to the specific form of the problem, it is computationally feasible to find the global minimum of ρ on a small subspace of \mathbb{V} . This allows to enrich the gradient direction with additional directions in order to accelerate convergence. Several strategies of this type exist of which LOBPCG and Jacob–Davidson have been extended to low-rank matrices and tensors. In particular, thanks to the multilinear structure of the TT format, it is feasible to minimize globally over a subspace in one of the TT cores. Proceeding in a sweeping manner, one can mimic the Jacob–Davidson method to TT tensors; see [91, 92].

9.5 Initial Value Problems

Instead of approximating only a single (very large) matrix or tensor X by low rank, we now consider the task of approximating a *time-dependent* tensor $X(t)$ directly by a low-rank tensor $Y(t)$. The tensor $X(t)$ is either given explicitly, or more interesting, as the solution of an initial value problem (IVP)

$$\dot{X}(t) = F(X(t)), \quad X(t_0) = X_0 \in \mathbb{V}, \quad (9.48)$$

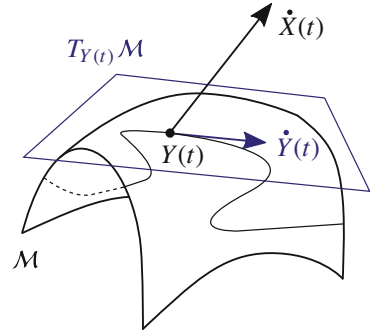
where \dot{X} means dX/dt . As it is usual, we assume that F is Lipschitz continuous with constant Λ ,

$$\|F(X) - F(Z)\| \leq \Lambda \|X - Z\| \quad \text{for all } X, Z \in \mathbb{V}, \quad (9.49)$$

so that the solution to (9.48) exists at least on some interval $[t_0, T]$. We took F autonomous, which can always be done by adding t as an extra integration parameter. For simplicity, we assume that the desired rank for the approximation $Y(t)$ is known and constant. In most applications, it will be important however that the numerical method that computes $Y(t)$ is robust to overestimation of the rank and/or allows for adapting the rank to improve the accuracy.

The aim is to obtain good approximations of $X(t)$ on the whole interval $[t_0, T]$. This is usually done by computing approximations $X_\ell \approx X(t_0 + \ell h)$ with h the time step. Classical time stepping methods for this include Runge–Kutta and BDF methods. Sometimes, one is only interested in the steady-state solution, that is, $X(t)$ for $t \rightarrow \infty$. This is for example the case for gradient flows, where F is the negative gradient of an objective function $f: \mathbb{V} \rightarrow \mathbb{R}$. The steady state solution of (9.48) is then a critical point of f , for example, a local minimizer. However, in such situations, it may be better to directly minimize f using methods from numerical optimization as explained in Sect. 9.4.

Fig. 9.7 Graphical depiction of the dynamical low-rank approximation $Y(t)$ at time t



9.5.1 Dynamical Low-Rank Approximation

We now explain how to obtain a low-rank approximation to (9.48) without needing to first solve for $X(t)$. Given an approximation submanifold $\mathcal{M} = \mathcal{M}_k$ or $\mathcal{M} = \mathcal{M}_k$ of fixed-rank matrices or tensors, the idea is to replace \dot{X} in (9.48) by the tangent vector in \mathcal{M} that is closest to $F(X)$; see also Fig. 9.7. It is easy to see that for the Frobenius norm, this tangent vector is $\mathcal{P}_X(F(X))$ where $\mathcal{P}_X: \mathbb{V} \rightarrow T_X\mathcal{M}$ is the orthogonal projection. Applying this substitution at every time t , we obtain a new IVP

$$\dot{Y}(t) = \mathcal{P}_{Y(t)}F(Y(t)), \quad Y(t_0) = Y_0 \in \mathcal{M}, \quad (9.50)$$

where $Y_0 = \mathcal{P}_{\mathcal{M}}(X_0)$ is a quasi-best approximation of X_0 in \mathcal{M} . In [59], the IVP (9.50) (or its solution) is aptly called the *dynamical low-rank approximation* (DLRA) of $X(t)$. Thanks to the tangent space projection, the solution $Y(t)$ will belong to \mathcal{M} as long as $\mathcal{P}_{Y(t)}$ exists, that is, until the rank of $Y(t)$ drops. In the following we assume that (9.50) can be integrated on $[t_0, T]$.

The DLRA (9.50) can equivalently be defined in weak form as follows: find, for each $t \in [t_0, T]$, an element $Y(t) \in \mathcal{M}$ such that

$$(\dot{Y}(t), Z)_F = (F(Y(t)), Z)_F \quad \text{for all } Z \in T_{Y(t)}\mathcal{M}, \quad (9.51)$$

and $Y(0) = Y_0 \in \mathcal{M}$. Observe that this can be seen as a time-dependent Galerkin condition since $T_{Y(t)}\mathcal{M}$ is a linear subspace that varies with t .

In the concrete case of low-rank matrices, DLRA appeared first in [59]. The same approximation principle, called *dynamically orthogonal* (DO), was also proposed in [94] for time-dependent stochastic PDEs. It was shown in [32, 83] that DO satisfies (9.50) after discretization of the stochastic and spatial domain. In theoretical physics, the *time-dependent variational principle* (TDVP) from [41] seems to be the first application of DLRA for simulating spin systems with uniform MPS, a variant of TT tensors. It is very likely similar ideas appeared well before since obtaining

approximations in a manifold from testing with tangent vectors as in (9.51) goes back as far as 1930 with the works of Dirac [22] and Frenkel [33]. We refer to [69] for a mathematical overview of this idea in quantum physics.

9.5.2 Approximation Properties

The local error at t of replacing (9.48) by (9.50) is minimized in Frobenius norm by the choice \dot{Y} ; see also Fig. 9.7. In order to quantify the effect of this approximation on the global error at the final time T , the simplest analysis is to assume as in [57, 58] that the vector field F is ε close to the tangent bundle of \mathcal{M} , that is,

$$\|F(Y(t)) - \mathcal{P}_{Y(t)}F(Y(t))\|_F \leq \varepsilon \quad \text{for all } t \in [t_0, T].$$

A simple comparison of IVPs then gives

$$\|Y(t) - X(t)\|_F \leq e^{\lambda t} \delta + (e^{\lambda t} - 1)\lambda^{-1}\varepsilon = O(\varepsilon + \delta), \quad (9.52)$$

where $\|X_0 - Y_0\|_F \leq \delta$ and λ is a *one-sided* Lipschitz constant of F satisfying⁸

$$(X - Z, F(X) - F(Z))_F \leq \lambda \|X - Z\|_F^2 \quad \text{for all } X, Z \in \mathbb{V}.$$

From (9.52), we observe that $Y(t)$ is guaranteed to be a good approximation of $X(t)$ but only for (relatively) short time intervals when $\lambda > 0$.

Alternatively, one can compare $Y(t)$ with a quasi-best approximation $Y_{\text{qb}}(t) \in \mathcal{M}$ to $X(t)$. Assuming $Y_{\text{qb}}(t)$ is continuously differentiable on $[t_0, T]$, this can be done by assuming that \mathcal{M} is not too curved along $Y_{\text{qb}}(t)$. In the matrix case, this means that the k th singular value of $Y_{\text{qb}}(t)$ is bounded from below, i.e., there exists $\rho > 0$ such that $\sigma_k(Y_{\text{qb}}(t)) \geq \rho$ for $t \in [t_0, T]$. Now a typical result from [59] is as follows: Let F be the identity operator and assume $\|\dot{X}(t)\|_F \leq c$ and $\|X(t) - Y_{\text{qb}}(t)\|_F \leq \rho/16$ for $t \in [t_0, T]$. Then,

$$\|Y(t) - Y_{\text{qb}}(t)\|_F \leq 2\beta e^{\beta t} \int_0^t \|Y_{\text{qb}}(s) - X(s)\|_F ds \quad \text{with } \beta = 8c\rho^{-1}$$

for $t_0 \leq t \leq \min(T, 0.55\beta^{-1})$. Hence, the approximation $Y(t)$ stays close to $Y_{\text{qb}}(t)$ for short times. We refer to [59] for additional results that also include the case of F not the identity. Most of the analysis was also extended to manifolds of fixed TT rank (as well as to Tucker and hierarchical) tensors in [60, 73] and to Hilbert spaces [83].

⁸We remark that λ can be negative and is bounded from above by Λ from (9.49).

We remark that these a-priori results only hold for (very) short times. In practice, they are overly pessimistic and in actual problems the accuracy is typically much higher than theoretically predicted; see [57, 71, 72, 83, 94], and the numerical example from Fig. 9.8 further below.

9.5.3 Low-Dimensional Evolution Equations

The dynamical low-rank problem (9.50) is an IVP that evolves on a manifold \mathcal{M} of fixed-rank matrices or tensors. In relevant applications, the rank will be small and hence we would like to integrate (9.50) by exploiting that \mathcal{M} has low dimension.

Let us explain how this is done for $m \times n$ matrices of rank k , that is, for the manifold \mathcal{M}_k . Then $\text{rank}(Y(t)) = k$ and we can write $Y(t) = U(t)S(t)V(t)^T$ where $U(t) \in \mathbb{R}^{m \times k}$ and $V(t) \in \mathbb{R}^{n \times k}$ have orthonormal columns and $S(t) \in \mathbb{R}^{k \times k}$. This is an SVD-like decomposition but we do not require $S(t)$ to be diagonal. The aim is now to formulate evolution equations for $U(t)$, $S(t)$, and $V(t)$.

To this end, recall from (9.10) that for fixed U, S, V every tangent vector \dot{Y} has a *unique* decomposition

$$\dot{Y} = \dot{U}SV^T + U\dot{S}V^T + US\dot{V}^T \quad \text{with } U^T\dot{U} = 0, V^T\dot{V} = 0.$$

Since $\dot{Y} = \mathcal{P}_Y(F(Y))$, we can isolate $\dot{U}, \dot{S}, \dot{V}$ by applying (9.12) with $Z = F(Y)$. The result is a new IVP equivalent to (9.50) but formulated in the factors:

$$\begin{cases} \dot{U} = (I - UU^T)F(Y)VS^{-1}, \\ \dot{S} = U^T F(Y)V, \\ \dot{V} = (I - VV^T)F(Y)^T US^{-T}. \end{cases} \quad (9.53)$$

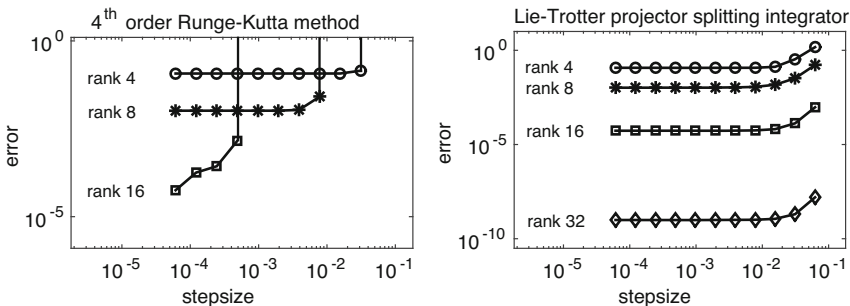


Fig. 9.8 Errors of the dynamical low-rank approximation for (9.55) integrated by a standard explicit Runge–Kutta scheme for (9.53) and the projector-splitting integrator (9.58). Picture taken from [58]. Copyright 2016 Society for Industrial and Applied Mathematics. Reprinted with permission. All rights reserved

(Here, U, S, V all depend on t .) Observe that this is a set of coupled non-linear ODEs, parametrized using $(m+n)k + k^2$ entries.

The ODE (9.53) appeared in [59, Prop. 2.1]. For DO [94], one uses the parametrization $Y(t) = U(t)M(t)^T$ where only U has orthonormal columns and obtains

$$\begin{cases} \dot{U} = (I - UU^T)F(UM^T)M(M^T M)^{-1}, \\ \dot{M} = F(UM^T)^T U. \end{cases} \quad (9.54)$$

These two coupled non-linear ODEs are very similar to (9.53) with respect to theoretical and numerical behavior. In particular, they also involve the normalization condition $U^T \dot{U} = 0$ and an explicit inverse $(M^T M)^{-1}$.

The derivation of these ODEs can be generalized to TT tensors with factored and gauged parametrizations for the tangent vectors. The equations are more tedious to write down explicitly, but relatively easy to implement. We refer to [73] for details. See also [4, 60] for application to the (hierarchical) Tucker tensor format.

For matrices and for tensors, the new IVPs have the advantage of being formulated in low dimensional parameters. However, they both suffer from a major problem: the time step in explicit methods needs to be in proportion to the smallest positive singular value of (each unfolding) of $Y(t)$. If these singular values become small (which is typically the case, since the DLRA approach by itself is reasonable for those applications where the true solution exhibits fast decaying singular values), Eq. (9.53) is very stiff. The presence of the terms S^{-1} in (9.53) and $(M^T M)^{-1}$ in (9.54) already suggests this and numerical experiments make this very clear. In Fig. 9.8, we report on the approximation errors for DLRA applied to the explicit time-dependent matrix

$$A(t) = \exp(tW_1) \exp(t) D \exp(tW_2), \quad 0 \leq t \leq 1, \quad (9.55)$$

with W_1, W_2 being skew-symmetric of size 100×100 and D a diagonal matrix with entries $2^{-1}, \dots, 2^{-100}$; see [58] for details. The left panel shows the results of a Runge–Kutta method applied to the resulting system (9.53). The method succeeds in computing a good low-rank approximation when the step size h is sufficiently small, but becomes unstable when h is larger than the smallest singular value of $Y(t)$. Due to this step-size restriction it hence becomes very expensive when aiming for accurate low-rank approximations. See also [57, Fig. 3] for similar results.

One solution would be to use expensive implicit methods or an ad-hoc regularization of S^{-1} . In the next subsection, a different approach is presented that is based on a splitting of the tangent space projector, and is robust to small singular values.

9.5.4 Projector-Splitting Integrator

Instead of immediately aiming for an ODE in the small factors U, S, V , the idea of the splitting integrator of [71] is to first apply a Lie splitting to the orthogonal projector \mathcal{P}_Y in

$$\dot{Y}(t) = \mathcal{P}_{Y(t)} F(Y(t)), \quad Y(t_0) = Y_0 \in \mathcal{M}, \quad (9.56)$$

and then—thanks to some serendipitous observation—obtain low dimensional ODEs at a later stage. For instance, in the matrix case, as stated in (9.9), the projector can be written as

$$\mathcal{P}_Y(Z) = ZVV^T - UU^TZVV^T + UU^TV \quad \text{with } Y = USV^T. \quad (9.57)$$

When we integrate each of these three terms consecutively (labeled a, b, c) from t_0 to $t_1 = t_0 + h$, we obtain the following scheme (all matrices depend on time):

$$\begin{cases} \dot{Y}_a = F(Y_a)V_aV_a^T, & Y_a(t_0) = Y_0 = U_aS_aV_a^T, \\ \dot{Y}_b = -U_bU_b^TF(Y_b)V_bV_b^T, & Y_b(t_0) = Y_a(t_1) = U_bS_bV_b^T, \\ \dot{Y}_c = U_cU_c^TF(Y_c), & Y_c(t_0) = Y_b(t_1) = U_cS_cV_c^T. \end{cases} \quad (9.58)$$

Here, all U_x and V_x are matrices with orthonormal columns. Observe the minus sign for Y_b . The result $Y_c(t_1)$ is an $O(h^2)$ approximation to $Y(t_1)$. We then repeat this scheme starting at $Y_c(t_1)$ and integrate from t_1 to $t_2 = t_1 + h$, and so on. By standard theory for Lie splittings, this scheme is first-order accurate for (9.56), that is, $\|Y_c(T) - Y(T)\|_F = O(h)$ where $T = \ell h$.

To integrate (9.58) we will first write it using much smaller matrices. To this end, observe that with exact integration $Y_a(t_1) \in \mathcal{M}_k$ since $\dot{Y}_a \in T_{Y_a}\mathcal{M}_k$ and $Y_a(t_0) \in \mathcal{M}_k$. Hence, we can substitute the ansatz $Y_a(t) = U_a(t)S_a(t)V_a(t)^T$ in the first substep and obtain

$$\dot{Y}_a = \frac{d}{dt}[U_a(t)S_a(t)]V_a(t)^T + U_a(t)S_a(t)\dot{V}_a(t)^T = F(Y_a(t))V_a(t)V_a(t)^T.$$

Judiciously choosing $\dot{V}_a(t) = 0$, we can simplify to

$$V_a(t) = V_a(t_0), \quad \frac{d}{dt}[U_a(t)S_a(t)] = F(Y_a(t))V_a(t_0).$$

Denoting $K(t) = U_a(t)S_a(t)$, the first substep is therefore equivalent to

$$\dot{K}(t) = F(K(t)V_a(t_0)^T)V_a(t_0), \quad K(t_0) = U_a(t_0)S(t_0). \quad (9.59)$$

Contrary to the earlier formulation, this is an IVP for an $n \times k$ matrix $K(t)$. The orthonormal matrix U_b for the next substep can be computed in $O(nk^2)$ work by a QR decomposition of $K(t_1)$.

The second and third substeps can be integrated analogously in terms of evolution equations only for $S_b(t)$ and $L_c(t) = V_c(t)S_c(t)$. Also note that we can take $V_b = V_a$ and $U_c = U_b$. We thus get a scheme, called KSL, that integrates in order K, S ,

and L . A second-order accurate scheme is the symmetric Strang splitting: one step consists of computing the K , S , L substeps for F with $h/2$ and afterwards the L , S , K substeps for the adjoint of F with $h/2$.

In both versions of the splitting scheme, care must be taken in the integration of the substeps since they are computationally the most expensive part. Fortunately, the ODEs in the substeps are formally of the same form as the original equation for the vector field $F(Y)$ since the projected subspace is constant; see, e.g., $V_a(t_0)$ in (9.59). This means that one can usually adapt specialized integrators for $F(Y)$. In [28], for example, the substeps arising from the Vlasov–Poisson equations in plasma physics (see also Sect. 9.6.5) can be integrated by spectral or semi-Lagrangian methods. In addition, when F is linear and has low TT matrix rank, the large matrix $K(t)V_a(t_0)^T$ in (9.59), for example, does not need to be formed explicitly when evaluating F . As illustration, for the Lyapunov operator $F(Z) = LZ + ZL^T$, the equation for K becomes

$$\dot{K}(t) = F(K(t)V_a(t_0)^T)V_a(t_0) = LK(t) + K(t)L_a, \quad L_a = V_a(t_0)^T L V_a(t_0)$$

where $L \in \mathbb{R}^{n \times n}$ is large but usually sparse, and $L_a \in \mathbb{R}^{k \times k}$ is small. Hence, an exponential integrator with a Krylov subspace method is ideally suited to integrate $K(t)$; see, e.g., [70].

Let us finish by summarizing some interesting properties of the splitting integrator for matrices. Let Y_ℓ be the solution after ℓ steps of the scheme explained above with step size h . For simplicity, we assume that each substep is solved exactly (or sufficiently accurately). Recall that $X(t)$ is the solution to the original ODE (9.48) that we approximate with the dynamical low-rank solution $Y(t)$ of (9.50).

- (a) *Exactness* [71, Thm. 4.1]: If the solution $X(t)$ of the original ODE lies on \mathcal{M}_k , then $Y_\ell = X(t_\ell)$ when F equals the identity.
- (b) *Robustness to small singular values* [58, Thm. 2.1]: There is no step size restriction due to small singular values. Concretely, under the same assumptions that lead to (9.52), the approximation error satisfies

$$\|Y_\ell - X(t_0 + \ell h)\|_F \leq C(\delta + \varepsilon + h) \quad \text{for all } \ell, h \text{ such that } t_0 + \ell h \leq T$$

where C only depends on F and T . Observe that this only introduced the time step error h . For the integration error $\|Y_\ell - Y(t_0 + \ell h)\|_F$, a similar bound exists in [71, Thm. 4.2] but it requires rather technical assumptions on F .

- (c) *Norm and energy conservation* [70, Lemma 6.3]: If the splitting scheme is applied to complex tensors for a Hamiltonian problem $F(X) = -i\mathcal{H}(X)$ with complex Hermitian \mathcal{H} , the Frobenius norm and energy are preserved:

$$\|Y_\ell\|_F = \|Y_0\|_F \quad \text{and} \quad \langle Y_\ell, \mathcal{H}(Y_\ell) \rangle_F = \langle Y_0, \mathcal{H}(Y_0) \rangle_F \quad \text{for all } n.$$

In the case of real tensors, the norm is preserved if $\langle F(Y), Y \rangle_F = 0$.

Property (a) does not seem very useful but it is key to showing the much more relevant property (b). All three properties are not shared when solving (9.53) by a standard integrator, like explicit Runge–Kutta. Even more, properties (a) and (b) are also lost for a different ordering of the splitting scheme, like KLS, even though that would still result in a first-order scheme. We remark that these properties also hold for the solution $Y(t)$ of the continuous problem by (formally) replacing h by 0.

To extend the idea of projector splitting to TT tensors $Y(t) \in \mathcal{M}_{\mathbf{k}}$, the correct splitting of the tangent space projector $\mathcal{P}_Y: \mathbb{V} \rightarrow T_Y \mathcal{M}_{\mathbf{k}}$ has to be determined. The idea in [72] is to take the sum expression (9.39) and split it as

$$\mathcal{P}_Y = \mathcal{P}_1^+ - \mathcal{P}_1^- + \mathcal{P}_2^+ - \mathcal{P}_2^- \cdots - \mathcal{P}_{d-1}^- + \mathcal{P}_d^+ \quad (9.60)$$

where

$$\mathcal{P}_\mu^+(Z) = \mathcal{P}_{\leq \mu-1}(\mathcal{P}_{\geq \mu+1}(Z)) \quad \text{and} \quad \mathcal{P}_\mu^-(Z) = \mathcal{P}_{\leq \mu}(\mathcal{P}_{\geq \mu+1}(Z)).$$

Observe that \mathcal{P}_μ^\pm depends on Y and that this splitting reduces to the matrix case in (9.57) when $d = 2$. The projector-splitting integrator for TT is now obtained by integrating each term in (9.60) from left to right:

$$\begin{cases} \dot{Y}_1^+ = \mathcal{P}_1^+(F(Y_1^+)), & Y_1^+(t_0) = Y_0, \\ \dot{Y}_1^- = -\mathcal{P}_1^-(F(Y_1^-)), & Y_1^-(t_0) = Y_1^+(t_1), \\ \dot{Y}_2^+ = \mathcal{P}_2^+(F(Y_2^+)), & Y_2^+(t_0) = Y_1^-(t_1), \\ \vdots & \vdots \\ \dot{Y}_d^+ = \mathcal{P}_d^+(F(Y_d^+)), & Y_d^+(t_0) = Y_{d-1}^-(t_1). \end{cases} \quad (9.61)$$

Quite remarkably, this splitting scheme for TT tensors shares many of the important properties from the matrix case. In particular, it allows for an efficient integration since only one core varies with time in each substep (see [72, Sec. 4]) and it is robust to small singular values in each unfolding (see [58, Thm. 3.1]). We refer to [42] for more details on its efficient implementation and its application to quantum spin systems in theoretical physics.

9.6 Applications

In this section, we explain different types of problems that have been solved by low-rank matrix and tensor methods in the literature. We will in particular focus on problems that can be approached by the geometry-oriented methods considered in this chapter, either via optimization on low-rank manifolds or via dynamical low-

rank integration. Our references to the literature are meant to give a broad and recent view of the usefulness of these methods, but we do not claim they are exhaustive.

9.6.1 Matrix Equations

In control and systems theory (see, e.g., [3]), a number of applications requires solving the following types of matrix equations:

$$\begin{aligned} \text{Lyapunov:} \quad & AX + XA^T = C, \\ \text{Sylvester:} \quad & AX + XB = C, \\ \text{Riccati:} \quad & AX + XA^T + XBX = C. \end{aligned} \tag{9.62}$$

Here, A , B , C are given matrices and X is the unknown matrix (of possible different size in each equation). The first two equations are linear, whereas the second is quadratic. For simplicity, we assume that these equations are uniquely solvable but there exist detailed results about conditions for this.

In large-scale applications, the matrix X is typically dense and too large to store. Under certain conditions, one can prove that X has fast decaying singular values and can thus be well approximated by a low-rank matrix; see [102] for an overview. For the linear equations, one can then directly attempt the optimization strategy explained in Sect. 9.4.2 and minimize the residual function or the energy-norm error. The latter is preferable but only possible when A and B are symmetric and positive definite; see [111] for a comparison. If the underlying matrices are ill-conditioned, as is the case with discretized PDEs, a simple Riemannian gradient scheme will not be effective and one needs to precondition the gradient steps or perform a quasi-Newton method. For example, in case of the Lyapunov equation, it is shown in [113] how to efficiently solve the Gauss–Newton equations for the manifold \mathcal{M}_k . If the Riccati equation is solved by Newton’s method, each step requires solving a Sylvester equation [102]. When aiming for low-rank approximations, the latter can again be solved by optimization on \mathcal{M}_k ; see [81]. We remark that while most methods for calculating low-rank approximations to (9.62) are based on Krylov subspaces and rational approximations, there exists a relation between both approaches; see [11].

The matrix equations from above have direct time-dependent versions. For example, the differential Riccati equation is given by

$$\dot{X}(t) = AX(t) + X(t)A^T + G(t, X(t)), \quad X(t_0) = X_0, \tag{9.63}$$

where $G(t, X(t)) = C - X(t)BX(t)$. Uniqueness of the solution $X(t)$ for all $t \geq t_0$ is guaranteed when X_0 , C , and B are symmetric and positive semi-definite [21]. In optimal control, the linear quadratic regulator problem with finite time horizon

requires solving (9.63). In the large-scale case, it is typical that X_0 and C are low rank and it has been observed [20, 76] that $X(t)$ has then fast decaying singular values, even on infinite time horizons.

Other examples are the differential Lyapunov equation ($G(t, X) = 0$) and the generalized differential Riccati equation ($G(t, X(t)) = C + \sum_{j=1}^J D_j^T X(t) D_j - X(t) B X(t)$); see, e.g. [20, 75], for applications. When matrices are large, it is important to exploit that applying the right hand side in (9.63) does not increase the rank of $X(t)$ too much, which is guaranteed here, if J is not too large and the matrix C is of low rank. In [89] a low-rank approximation to $X(t)$ is obtained with the dynamical low-rank algorithm. Like in the time-independent case, discretized PDEs might need special treatment to cope with the stiff ODEs. In particular, an exponential integrator can be combined with the projector-splitting integrator by means of an additional splitting of the vector field for the stiff part; see [89] for details and analysis.

9.6.2 Schrödinger Equation

Arguably the most typical example involving tensors of very high order is the time-dependent Schrödinger equation,

$$\dot{\psi} = -i\mathbf{H}(\psi),$$

where \mathbf{H} is a self-adjoint Hamiltonian operator acting on a (complex-valued) multi-particle wave function $\psi(x_1, \dots, x_d, t)$ with $x_\mu \in \mathbb{R}^p$, $p \leq 3$. This equation is fundamental in theoretical physics for the simulation of elementary particles and molecules. Employing a Galerkin discretization with $i = 1, \dots, n_\mu$ basis functions $\varphi_i^{(\mu)}$ in each mode $\mu = 1, \dots, d$, the wave function will be approximated as

$$\psi(x_1, \dots, x_d, t) \approx \sum_{i_1}^{n_1} \cdots \sum_{i_d}^{n_d} X(i_1, \dots, i_d; t) \varphi_{i_1}^{(1)}(x_1) \cdots \varphi_{i_d}^{(d)}(x_d).$$

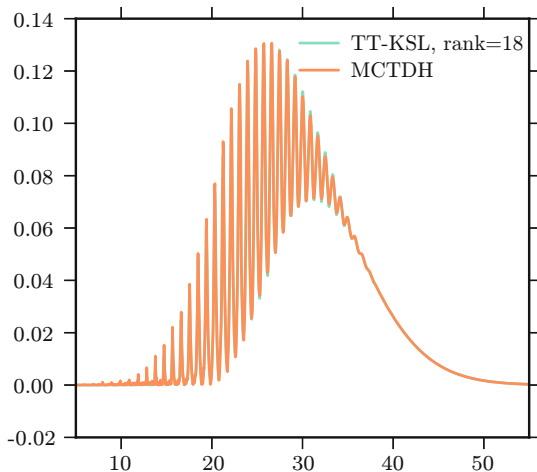
By regarding the unknown complex coefficient $X(i_1, \dots, i_d; t)$ as the (i_1, \dots, i_d) th element of the time-dependent tensor $X(t)$ of size $n_1 \times \cdots \times n_d$, we obtain the linear differential equation

$$\dot{X}(t) = -i\mathcal{H}(X(t)) \tag{9.64}$$

where \mathcal{H} is the Galerkin discretization of the Hamiltonian \mathbf{H} . More complicated versions of this equation allow the Hamiltonian to be time-dependent.

The size of the tensor $X(t)$ will be unmanageable for large d but, fortunately, certain systems allow it to be approximated by a low-rank tensor. For example,

Fig. 9.9 Spectrum computed by the second-order splitting integrator and by the MCTDH package. Picture taken from [72]. Copyright 2015 Society for Industrial and Applied Mathematics. Reprinted with permission. All rights reserved



in the multilayer multiconfiguration time-dependent Hartree model (ML-MCDTH) in [78, 116] for simulating quantum dynamics in small molecules, the wave functions are approximated by hierarchical Tucker tensors. Spin systems in theoretical physics, on the other hand, employ the TT format and can simulate systems of very large dimension (since n_μ are small); see [97] for an overview. For both application domains, the solution of (9.64) can be obtained by applying dynamical low-rank; see, e.g., [77] for MCDTH and [41] for spin systems.

Numerical experiments for (9.64) with the Henon–Heiles potential were performed in [72]. There the second-order splitting integrator with a fixed time step $h = 0.01$ and a fixed TT rank of 18 was compared to an adaptive integration of the gauged ODEs, similar to (9.53). In particular, the ML-MCDTH method [10] was used in the form of the state-of-the-art code `mcdth v8.4`. Except for the slightly different tensor formats (TT versus hierarchical Tucker) all other modeling parameters are the same. For similar accuracy, a 10 dimensional problem is integrated by `mcdth` in 54 354 s, whereas the TT splitting integrator required only 4425 s. The reason for this time difference was mainly due to the ill conditioned ODEs in the gauged representation. In addition, there was no visible difference in the Fourier transform of the auto-correlation functions; see Fig. 9.9.

There is an interesting link between the computation of the ground state (eigenvector of the minimal eigenvalue) of \mathcal{H} via the minimization of the Rayleigh quotient

$$\rho(X) = \frac{(X, \mathcal{H}(X))_F}{(X, X)_F}$$

and so-called *imaginary time evolution* [41] for a scaled version of (9.64) that conserves unit norm. The latter is a formal way to obtain a gradient flow for $\rho(X)$ using imaginary time $\tau = -it$ by integrating

$$\dot{X} = -\mathcal{H}(X) + (X, \mathcal{H}(X))_F X.$$

For both approaches, we can approximate their solutions with low-rank tensors, as we explained before, either via optimization or dynamical low rank on \mathcal{M}_k . However, methods based on optimization of the multilinear TT representation of X remain the more popular approach since they easily allow to reuse certain techniques from standard eigenvalue problems, like subspace corrections, as is done in the DMRG [118] or AMEn [23, 63] algorithm.

For an overview on tensor methods in quantum physics and chemistry we refer to [51, 97, 105].

9.6.3 Matrix and Tensor Completion

Let $\Omega \subset \{1, \dots, m\} \times \{1, \dots, n\}$ be a sampling set. The problem of matrix completion consists of recovering an unknown $m \times n$ matrix M of rank k based only on the values $M(i, j)$ for all $(i, j) \in \Omega$. Remarkably, this problem has a unique solution if $|\Omega| \approx O(\dim \mathcal{M}_k) = O(k(m+n))$ and under certain randomness conditions on Ω and M ; see [14]. If the rank k is known, this suggests immediately the strategy of recovering M by minimizing the least-squares fit

$$f(X) = \sum_{i=1}^m \sum_{j=1}^n (X(i, j) - M(i, j))^2 = \|\mathcal{P}_\Omega(X - M)\|_F^2$$

on the manifold \mathcal{M}_k , where \mathcal{P}_Ω is the orthogonal projection onto matrices that vanish outside of Ω . Since \mathcal{P}_Ω is well-conditioned on \mathcal{M}_k when the iterates satisfy an incoherence property, the simple Riemannian gradient schemes that we explained above perform very well in recovering M ; see, e.g., [82, 112].

The problem of matrix completion can be generalized to tensors, and Riemannian methods for tensor completion have been developed for the Tucker format in [64], and for the TT format in [103].

In addition, instead of element-wise sampling, the observations can also be constructed from a general linear operator $\mathcal{S}: \mathbb{V} \rightarrow \mathbb{R}^q$. This problem remains well-posed under certain randomness conditions on \mathcal{S} and also Riemannian optimization performs well if applied to the least-square version of the problem for which $\mathcal{L} = \mathcal{S}^T \mathcal{S}$; see [117].

9.6.4 Stochastic and Parametric Equations

Other interesting applications for low-rank tensors arise from stochastic or parametric PDEs [7, 9, 24, 26, 31, 54, 55]. For simplicity, suppose that the system matrix of

a finite-dimensional linear system $Ax = b$ of equations depends on p parameters $\omega^{(1)}, \dots, \omega^{(p)}$, that is,

$$A(\omega^{(1)}, \dots, \omega^{(p)})x(\omega^{(1)}, \dots, \omega^{(p)}) = b. \quad (9.65)$$

One might be interested in the solution $x \in \mathbb{R}^n$ for some or all choices of parameters, or, in case the parameters are random variables, in expectation values of certain quantities of interest.

By discretizing each parameter $\omega^{(\mu)}$ with m_μ values, we can gather all the $m_1 \cdots m_p$ solution vectors x into one tensor

$$[X(j, i_1, \dots, i_p)] = [x_j(\omega_{i_1}^{(1)}, \dots, \omega_{i_p}^{(p)})]$$

of order $p + 1$ and size $n \times m_1 \times \cdots \times m_p$. When A depends analytically on $\omega = (\omega^{(1)}, \dots, \omega^{(p)})$, the tensor X can be shown [62] to be well approximated by low TT rank and it satisfies a very large linear system $\mathcal{L}(X) = B$. If \mathcal{L} is a TT matrix of low rank, we can then approximate X on \mathcal{M}_k by the optimization techniques we discussed in Sect. 9.4. This is done, for example, in [65] with an additional preconditioning of the gradient.

A similar situation arises for elliptic PDEs with stochastic coefficients. After truncated Karhunen–Loève expansion, one can obtain a deterministic PDE of the same form as (9.65); see [106]. The following time-dependent example with random variable ω ,

$$\partial_t \psi + v(t, x; \omega) \cdot \nabla \psi = 0, \quad \psi(0, x) = x,$$

was solved by dynamical low-rank in [32].

9.6.5 Transport Equations

Transport equations describe (densities of) particles at position $x \in \mathbb{R}^p$ and velocity $v \in \mathbb{R}^p$. They are typically more challenging to integrate than purely diffusive problems. For example, the Vlasov equation

$$\partial_t u(t, x, v) + v \cdot \nabla_x u(t, x, v) - F(u) \cdot \nabla_v u(t, x, v) = 0 \quad (9.66)$$

is a kinetic model for the density u of electrons in plasma. The function F is a nonlinear term representing the force. These equations can furthermore be coupled with Maxwell's equations resulting in systems that require specialized integrators to preserve conservation laws in the numerical solution. After spatial discretization on a tensor product grid, Eq. (9.66) becomes a differential equation for a large tensor of order $d = 6$. In the case of the Vlasov–Poisson and Vlasov–Maxwell equations, [28,

30] show the splitting integrator gives very good approximations with modest TT rank, even over relatively large time intervals. In addition, the numerical integration of the substeps can be modified to ensure better preservation of some conservation laws; see [29, 30].

Similar approaches appear for weakly compressible fluid flow with the Boltzmann equation in [27] and stochastic transport PDEs in [32]. The latter also shows that numerical filters can be used in combination with dynamical low-rank to successfully reduce artificial oscillations.

9.7 Conclusions

In this chapter we have shown how the geometry of low-rank matrices and TT tensors can be exploited in algorithms. We focused on two types of problems: Riemannian optimization for solving large linear systems and eigenvalue problems, and dynamical low-rank approximation for initial value problems. Our aim was to be sufficiently explanatory without sacrificing readability and we encourage the interested reader to refer to the provided references for a more in depth treatment of these subjects.

Several things have not been discussed in this introductory chapter. The most important issue is arguably the rank adaptation during the course of the algorithms to match the desired tolerance at convergence. For this, truncation of singular values with a target error instead of a target rank can be used both for matrices and TT tensors, but from a conceptual perspective such an approach is at odds with algorithms that are defined on manifolds of fixed rank matrices or tensors. However, it is possible to combine geometric methods with rank adaptivity as in [109] for greedy rank-one optimization and in [42] for a two-site version of the splitting scheme for time integration, yet many theoretical and implementation questions remain. Other important topics not covered are the problem classes admitting a priori low-rank approximability [19, 44], the application of low-rank formats to seemingly non-high dimensional problems like quantized TT (QTT) [52, 53], the efficient numerical implementation of truly large-scale and stiff problems, schemes with guaranteed and optimal convergence as in [5], and more general tensor networks like PEPS [114].

References

1. Absil, P.A., Oseledets, I.V.: Low-rank retractions: a survey and new results. *Comput. Optim. Appl.* **62**(1), 5–29 (2015)
2. Absil, P.A., Mahony, R., Sepulchre, R.: *Optimization Algorithms on Matrix Manifolds*. Princeton University Press, Princeton (2008)
3. Antoulas, A.C.: *Approximation of Large-Scale Dynamical Systems*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia (2005)

4. Arnold, A., Jahnke, T.: On the approximation of high-dimensional differential equations in the hierarchical Tucker format. *BIT* **54**(2), 305–341 (2014)
5. Bachmayr, M., Dahmen, W.: Adaptive near-optimal rank tensor approximation for high-dimensional operator equations. *Found. Comput. Math.* **15**(4), 839–898 (2015)
6. Bachmayr, M., Schneider, R., Uschmajew, A.: Tensor networks and hierarchical tensors for the solution of high-dimensional partial differential equations. *Found. Comput. Math.* **16**(6), 1423–1472 (2016)
7. Bachmayr, M., Cohen, A., Dahmen, W.: Parametric PDEs: sparse or low-rank approximations? *IMA J. Numer. Anal.* **38**(4), 1661–1708 (2018)
8. Ballani, J., Grasedyck, L.: A projection method to solve linear systems in tensor format. *Numer. Linear Algebra Appl.* **20**(1), 27–43 (2013)
9. Ballani, J., Grasedyck, L.: Hierarchical tensor approximation of output quantities of parameter-dependent PDEs. *SIAM/ASA J. Uncertain. Quantif.* **3**(1), 852–872 (2015)
10. Beck, M.H., Jäckle, A., Worth, G.A., Meyer, H.D.: The multiconfiguration time-dependent Hartree (MCTDH) method: a highly efficient algorithm for propagating wavepackets. *Phys. Rep.* **324**(1), 1–105 (2000)
11. Benner, P., Breiten, T.: On optimality of approximate low rank solutions of large-scale matrix equations. *Syst. Control Lett.* **67**, 55–64 (2014)
12. Boumal, N., Absil, P.A., Cartis, C.: Global rates of convergence for nonconvex optimization on manifolds. *IMA J. Numer. Anal.* **39**(1), 1–33 (2019)
13. Breiding, P., Vannieuwenhoven, N.: A Riemannian trust region method for the canonical tensor rank approximation problem. *SIAM J. Optim.* **28**(3), 2435–2465 (2018)
14. Candès, E.J., Tao, T.: The power of convex relaxation: near-optimal matrix completion. *IEEE Trans. Inform. Theory* **56**(5), 2053–2080 (2010)
15. Cichocki, A., Mandic, D., De Lathauwer, L., Zhou, G., Zhao, Q., Caiifa, C., Phan, H.A.: Tensor decompositions for signal processing applications: from two-way to multiway component analysis. *IEEE Signal Proc. Mag.* **32**(2), 145–163 (2015)
16. Cichocki, A., Lee, N., Oseledets, I., Phan, A.H., Zhao, Q., Mandic, D.P.: Tensor networks for dimensionality reduction and large-scale optimization. Part 1: low-rank tensor decompositions. *Found. Trends Mach. Learn.* **9**(4–5), 249–429 (2016)
17. Cichocki, A., Phan, A.H., Zhao, Q., Lee, N., Oseledets, I., Sugiyama, M., Mandic, D.P.: Tensor networks for dimensionality reduction and large-scale optimization. Part 2: applications and future perspectives. *Found. Trends Mach. Learn.* **9**(6), 431–673 (2017)
18. Criscitiello, C., Boumal, N.: Efficiently escaping saddle points on manifolds (2019). [arXiv:1906.04321](https://arxiv.org/abs/1906.04321)
19. Dahmen, W., DeVore, R., Grasedyck, L., Süli, E.: Tensor-sparsity of solutions to high-dimensional elliptic partial differential equations. *Found. Comput. Math.* **16**(4), 813–874 (2016)
20. Damm, T., Mena, H., Stillfjord, T.: Numerical solution of the finite horizon stochastic linear quadratic control problem. *Numer. Linear Algebra Appl.* **24**(4), e2091, 11 (2017)
21. Dieci, L., Eirola, T.: Positive definiteness in the numerical solution of Riccati differential equations. *Numer. Math.* **67**(3), 303–313 (1994)
22. Dirac, P.A.M.: Note on exchange phenomena in the Thomas atom. *Proc. Camb. Philos. Soc.* **26**, 376–385 (1930)
23. Dolgov, S.V., Savostyanov, D.V.: Alternating minimal energy methods for linear systems in higher dimensions. *SIAM J. Sci. Comput.* **36**(5), A2248–A2271 (2014)
24. Dolgov, S., Khoromskij, B.N., Litvinenko, A., Matthies, H.G.: Polynomial chaos expansion of random coefficients and the solution of stochastic partial differential equations in the tensor train format. *SIAM/ASA J. Uncertain. Quantif.* **3**(1), 1109–1135 (2015)
25. Eckart, C., Young, G.: The approximation of one matrix by another of lower rank. *Psychometrika* **1**(3), 211–218 (1936)
26. Eigel, M., Pfeffer, M., Schneider, R.: Adaptive stochastic Galerkin FEM with hierarchical tensor representations. *Numer. Math.* **136**(3), 765–803 (2017)

27. Einkemmer, L.: A low-rank algorithm for weakly compressible flow. *SIAM J. Sci. Comput.* **41**(5), A2795–A2814 (2019)
28. Einkemmer, L., Lubich, C.: A low-rank projector-splitting integrator for the Vlasov–Poisson equation. *SIAM J. Sci. Comput.* **40**(5), B1330–B1360 (2018)
29. Einkemmer, L., Lubich, C.: A quasi-conservative dynamical low-rank algorithm for the Vlasov equation. *SIAM J. Sci. Comput.* **41**(5), B1061–B1081(2019)
30. Einkemmer, L., Ostermann, A., Piazzola, C.: A low-rank projector-splitting integrator for the Vlasov–Maxwell equations with divergence correction (2019). arXiv:1902.00424
31. Espig, M., Hackbusch, W., Litvinenko, A., Matthies, H.G., Wähnert, P.: Efficient low-rank approximation of the stochastic Galerkin matrix in tensor formats. *Comput. Math. Appl.* **67**(4), 818–829 (2014)
32. Feppon, F., Lermusiaux, P.F.J.: A geometric approach to dynamical model order reduction. *SIAM J. Matrix Anal. Appl.* **39**(1), 510–538 (2018)
33. Frenkel, J.: *Wave Mechanics: Advanced General Theory*. Clarendon Press, Oxford (1934)
34. Golub, G., Kahan, W.: Calculating the singular values and pseudo-inverse of a matrix. *SIAM J. Numer. Anal.* **2**(2), 205–224 (1965)
35. Golub, G.H., Van Loan, C.F.: *Matrix Computations*, 4th edn. Johns Hopkins University Press, Baltimore (2013)
36. Grasedyck, L.: Hierarchical singular value decomposition of tensors. *SIAM J. Matrix Anal. Appl.* **31**(4), 2029–2054 (2010)
37. Grasedyck, L., Kressner, D., Tobler, C.: A literature survey of low-rank tensor approximation techniques. *GAMM-Mitt.* **36**(1), 53–78 (2013)
38. Grohs, P., Hosseini, S.: Nonsmooth trust region algorithms for locally Lipschitz functions on Riemannian manifolds. *IMA J. Numer. Anal.* **36**(3), 1167–1192 (2016)
39. Hackbusch, W.: *Tensor Spaces and Numerical Tensor Calculus*. Springer, Heidelberg (2012)
40. Hackbusch, W., Kühn, S.: A new scheme for the tensor representation. *J. Fourier Anal. Appl.* **15**(5), 706–722 (2009)
41. Haegeman, J., Cirac, I., Osborne, T., Pižorn, I., Verschelde, H., Verstraete, F.: Time-dependent variational principle for quantum lattices. *Phys. Rev. Lett.* **107**(7), 070601 (2011)
42. Haegeman, J., Lubich, C., Oseledets, I., Vandereycken, B., Verstraete, F.: Unifying time evolution and optimization with matrix product states. *Phys. Rev. B* **94**(16), 165116 (2016)
43. Halko, N., Martinsson, P.G., Tropp, J.A.: Finding structure with randomness: probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Rev.* **53**(2), 217–288 (2011)
44. Hastings, M.B.: An area law for one-dimensional quantum systems. *J. Stat. Mech. Theory Exp.* **2007**, P08024 (2007)
45. Helmke, U., Shayman, M.A.: Critical points of matrix least squares distance functions. *Linear Algebra Appl.* **215**, 1–19 (1995)
46. Holtz, S., Rohwedder, T., Schneider, R.: On manifolds of tensors of fixed TT-rank. *Numer. Math.* **120**(4), 701–731 (2012)
47. Hosseini, S., Uschmajew, A.: A Riemannian gradient sampling algorithm for nonsmooth optimization on manifolds. *SIAM J. Optim.* **27**(1), 173–189 (2017)
48. Hosseini, S., Huang, W., Yousefpour, R.: Line search algorithms for locally Lipschitz functions on Riemannian manifolds. *SIAM J. Optim.* **28**(1), 596–619 (2018)
49. Jain, P., Meka, R., Dhillon, I.S.: Guaranteed rank minimization via singular value projection. In: *Advances in Neural Information Processing Systems*, vol. 23, pp. 937–945 (2010)
50. Kazeev, V.A., Khoromskij, B.N.: Low-rank explicit QTT representation of the Laplace operator and its inverse. *SIAM J. Matrix Anal. Appl.* **33**(3), 742–758 (2012)
51. Khoromskaya, V., Khoromskij, B.N.: *Tensor Numerical Methods in Quantum Chemistry*. De Gruyter, Berlin (2018)
52. Khoromskij, B.N.: $O(d \log N)$ -quantics approximation of N - d tensors in high-dimensional numerical modeling. *Constr. Approx.* **34**(2), 257–280 (2011)
53. Khoromskij, B.N.: *Tensor Numerical Methods in Scientific Computing*. De Gruyter, Berlin (2018)

54. Khoromskij, B.N., Oseledets, I.: Quantics-TT collocation approximation of parameter-dependent and stochastic elliptic PDEs. *Comput. Methods Appl. Math.* **10**(4), 376–394 (2010)
55. Khoromskij, B.N., Schwab, C.: Tensor-structured Galerkin approximation of parametric and stochastic elliptic PDEs. *SIAM J. Sci. Comput.* **33**(1), 364–385 (2011)
56. Khoromskij, B.N., Oseledets, I.V., Schneider, R.: Efficient time-stepping scheme for dynamics on TT-manifolds (2012). MPI MiS Preprint 24/2012
57. Kieri, E., Vandereycken, B.: Projection methods for dynamical low-rank approximation of high-dimensional problems. *Comput. Methods Appl. Math.* **19**(1), 73–92 (2019)
58. Kieri, E., Lubich, C., Walach, H.: Discretized dynamical low-rank approximation in the presence of small singular values. *SIAM J. Numer. Anal.* **54**(2), 1020–1038 (2016)
59. Koch, O., Lubich, C.: Dynamical low-rank approximation. *SIAM J. Matrix Anal. Appl.* **29**(2), 434–454 (2007)
60. Koch, O., Lubich, C.: Dynamical tensor approximation. *SIAM J. Matrix Anal. Appl.* **31**(5), 2360–2375 (2010)
61. Kolda, T.G., Bader, B.W.: Tensor decompositions and applications. *SIAM Rev.* **51**(3), 455–500 (2009)
62. Kressner, D., Tobler, C.: Low-rank tensor Krylov subspace methods for parametrized linear systems. *SIAM J. Matrix Anal. Appl.* **32**(4) (2011)
63. Kressner, D., Steinlechner, M., Uschmajew, A.: Low-rank tensor methods with subspace correction for symmetric eigenvalue problems. *SIAM J. Sci. Comput.* **36**(5), A2346–A2368 (2014)
64. Kressner, D., Steinlechner, M., Vandereycken, B.: Low-rank tensor completion by Riemannian optimization. *BIT* **54**(2), 447–468 (2014)
65. Kressner, D., Steinlechner, M., Vandereycken, B.: Preconditioned low-rank Riemannian optimization for linear systems with tensor product structure. *SIAM J. Sci. Comput.* **38**(4), A2018–A2044 (2016)
66. Lee, J.M.: *Introduction to Smooth Manifolds*. Springer, New York (2003)
67. Lehoucq, R.B., Sorensen, D.C., Yang, C.: *ARPACK users’ guide: solution of large-scale eigenvalue problems with implicitly restarted Arnoldi methods*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia (1998)
68. Lewis, A.S., Malick, J.: Alternating projections on manifolds. *Math. Oper. Res.* **33**(1), 216–234 (2008)
69. Lubich, C.: *From Quantum to Classical Molecular Dynamics: Reduced Models and Numerical Analysis*. European Mathematical Society (EMS), Zürich (2008)
70. Lubich, C.: Time integration in the multiconfiguration time-dependent Hartree method of molecular quantum dynamics. *Appl. Math. Res. Express. AMRX* **2015**(2), 311–328 (2015)
71. Lubich, C., Oseledets, I.: A projector-splitting integrator for dynamical low-rank approximation. *BIT* **54**(1), 171–188 (2014)
72. Lubich, C., Oseledets, I., Vandereycken, B.: Time integration of tensor trains. *SIAM J. Numer. Anal.* **53**(2), 917–941 (2015)
73. Lubich, C., Rohwedder, T., Schneider, R., Vandereycken, B.: Dynamical approximation of hierarchical Tucker and tensor-train tensors. *SIAM J. Matrix Anal. Appl.* **34**(2), 470–494 (2013)
74. Luenberger, D.G.: The gradient projection method along geodesics. *Manage. Sci.* **18**, 620–631 (1972)
75. Mena, H., Pfurtscheller, L.: An efficient SPDE approach for El Niño. *Appl. Math. Comput.* **352**, 146–156 (2019)
76. Mena, H., Ostermann, A., Pfurtscheller, L.M., Piazzola, C.: Numerical low-rank approximation of matrix differential equations. *J. Comput. Appl. Math.* **340**, 602–614 (2018)
77. Meyer, H.D.: Studying molecular quantum dynamics with the multiconfiguration time-dependent Hartree method. *Wiley Interdiscip. Rev. Comput. Mol. Sci.* **2**(2), 351–374 (2012)
78. Meyer, H., Manthea, U., Cederbaum, L.S.: The multi-configurational time-dependent Hartree approach. *Chem. Phys. Lett.* **165**(1), 73–78 (1990)

79. Meyer, G., Journée, M., Bonnabel, S., Sepulchre, R.: From subspace learning to distance learning: a geometrical optimization approach. In: Proceedings of the IEEE/SP 15th Workshop on Statistical Signal Processing, pp. 385–388 (2009)
80. Mirsky, L.: Symmetric gauge functions and unitarily invariant norms. *Quart. J. Math. Oxf. Ser. (2)* **11**, 50–59 (1960)
81. Mishra, B., Vandereycken, B.: A Riemannian approach to low-rank Algebraic Riccati equations. In: 21st International Symposium on Mathematical Theory of Networks and Systems, pp. 965–968 (2014)
82. Mishra, B., Meyer, G., Bonnabel, S., Sepulchre, R.: Fixed-rank matrix factorizations and Riemannian low-rank optimization. *Comput. Stat.* **29**(3–4), 591–621 (2014)
83. Musharbash, E., Nobile, F., Zhou, T.: Error analysis of the dynamically orthogonal approximation of time dependent random PDEs. *SIAM J. Sci. Comput.* **37**(3), A776–A810 (2015)
84. Orsi, R., Helmke, U., Moore, J.B.: A Newton-like method for solving rank constrained linear matrix inequalities. In: Proceedings of the 43rd IEEE Conference on Decision and Control, pp. 3138–3144 (2004)
85. Oseledets, I.V.: Approximation of $2^d \times 2^d$ matrices using tensor decomposition. *SIAM J. Matrix Anal. Appl.* **31**(4), 2130–2145 (2010)
86. Oseledets, I.V.: Tensor-train decomposition. *SIAM J. Sci. Comput.* **33**(5), 2295–2317 (2011)
87. Oseledets, I.V., Tyrtshnikov, E.E.: Breaking the curse of dimensionality, or how to use SVD in many dimensions. *SIAM J. Sci. Comput.* **31**(5), 3744–3759 (2009)
88. Oseledets, I., Tyrtshnikov, E.: TT-cross approximation for multidimensional arrays. *Linear Algebra Appl.* **432**(1), 70–88 (2010)
89. Ostermann, A., Piazzola, C., Walach, H.: Convergence of a low-rank Lie-Trotter splitting for stiff matrix differential equations. *SIAM J. Numer. Anal.* **57**(4), 1947–1966 (2019)
90. Park, D., Kyriillidis, A., Carmanis, C., Sanghavi, S.: Non-square matrix sensing without spurious local minima via the Burer-Monteiro approach. In: Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, pp. 65–74 (2017)
91. Rakhuba, M.V., Oseledets, I.V.: Jacobi-Davidson method on low-rank matrix manifolds. *SIAM J. Sci. Comput.* **40**(2), A1149–A1170 (2018)
92. Rakhuba, M., Novikov, A., Oseledets, I.: Low-rank Riemannian eigensolver for high-dimensional Hamiltonians. *J. Comput. Phys.* **396**, 718–737 (2019)
93. Rauhut, H., Schneider, R., Stojanac, Ž.: Low rank tensor recovery via iterative hard thresholding. *Linear Algebra Appl.* **523**, 220–262 (2017)
94. Sapsis, T.P., Lermusiaux, P.F.J.: Dynamically orthogonal field equations for continuous stochastic dynamical systems. *Phys. D* **238**(23–24), 2347–2360 (2009)
95. Sato, H., Kasai, H., Mishra, B.: Riemannian stochastic variance reduced gradient algorithm with retraction and vector transport. *SIAM J. Optim.* **29**(2), 1444–1472 (2019)
96. Schmidt, E.: Zur Theorie der linearen und nichtlinearen Integralgleichungen. *Math. Ann.* **63**(4), 433–476 (1907)
97. Schollwöck, U.: The density-matrix renormalization group in the age of matrix product states. *Ann. Phys.* **326**(1), 96–192 (2011)
98. Shalit, U., Weinshall, D., Chechik, G.: Online learning in the manifold of low-rank matrices. In: Advances in Neural Information Processing Systems, vol. 23, pp. 2128–2136 (2010)
99. Shub, M.: Some remarks on dynamical systems and numerical analysis. In: Dynamical systems and partial differential equations (Caracas, 1984), pp. 69–91. University Simon Bolivar, Caracas (1986)
100. Sidiropoulos, N.D., De Lathauwer, L., Fu, X., Huang, K., Papalexakis, E.E., Faloutsos, C.: Tensor decomposition for signal processing and machine learning. *IEEE Trans. Signal Process.* **65**(13), 3551–3582 (2017)
101. Signoretto, M., Tran Dinh, Q., De Lathauwer, L., Suykens, J.A.K.: Learning with tensors: a framework based on convex optimization and spectral regularization. *Mach. Learn.* **94**(3), 303–351 (2014)
102. Simoncini, V.: Computational methods for linear matrix equations. *SIAM Rev.* **58**(3), 377–441 (2016)

103. Steinlechner, M.: Riemannian optimization for high-dimensional tensor completion. *SIAM J. Sci. Comput.* **38**(5), S461–S484 (2016)
104. Stewart, G.W.: On the early history of the singular value decomposition. *SIAM Rev.* **35**(4), 551–566 (1993)
105. Szalay, S., Pfeiffer, M., Murg, V., Barcza, G., Verstraete, F., Schneider, R., Legeza, O.: Tensor product methods and entanglement optimization for ab initio quantum chemistry. *Int. J. Quantum Chem.* **115**(19), 1342–1391 (2015)
106. Todor, R.A., Schwab, C.: Convergence rates for sparse chaos approximations of elliptic problems with stochastic coefficients. *IMA J. Numer. Anal.* **27**(2), 232–261 (2007)
107. Udriște, C.: Convex functions and optimization methods on Riemannian manifolds. Kluwer Academic Publishers Group, Dordrecht (1994)
108. Uschmajew, A., Vandereycken, B.: The geometry of algorithms using hierarchical tensors. *Linear Algebra Appl.* **439**(1), 133–166 (2013)
109. Uschmajew, A., Vandereycken, B.: Greedy rank updates combined with Riemannian descent methods for low-rank optimization. In: 2015 International Conference on Sampling Theory and Applications (SampTA), pp. 420–424 (2015)
110. Uschmajew, A., Vandereycken, B.: On critical points of quadratic low-rank matrix optimization problems (2018). MPI MiS Preprint 58/2018
111. Vandereycken, B.: Riemannian and multilevel optimization for rank-constrained matrix problems. Ph.D. thesis, Department of Computer Science, KU Leuven (2010)
112. Vandereycken, B.: Low-rank matrix completion by Riemannian optimization. *SIAM J. Optim.* **23**(2), 1214–1236 (2013)
113. Vandereycken, B., Vandewalle, S.: A Riemannian optimization approach for computing low-rank solutions of Lyapunov equations. *SIAM J. Matrix Anal. Appl.* **31**(5), 2553–2579 (2010)
114. Verstraete, F., Cirac, J.I.: Renormalization algorithms for quantum-many body systems in two and higher dimensions (2004). arXiv:cond-mat/0407066
115. Verstraete, F., García-Ripoll, J.J., Cirac, J.I.: Matrix product density operators: simulation of finite-temperature and dissipative systems. *Phys. Rev. Lett.* **93**(20), 207204 (2004)
116. Wang, H., Thoss, M.: Multilayer formulation of the multiconfiguration time-dependent Hartree theory. *J. Chem. Phys.* **119**(3), 1289–1299 (2003)
117. Wei, K., Cai, J.F., Chan, T.F., Leung, S.: Guarantees of Riemannian optimization for low rank matrix recovery. *SIAM J. Matrix Anal. Appl.* **37**(3), 1198–1222 (2016)
118. White, S.R.: Density-matrix algorithms for quantum renormalization groups. *Phys. Rev. B* **48**(14), 10345 (1993)

Open Access This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.

